UNIVERSIDAD PÚBLICA DE EL ALTO

CARRERA DE INGENIERÍA DE SISTEMAS

PROYECTO DE GRADO

“SISTEMA WEB DE SEGUIMIENTO INDIVIDUAL UNIVERSITARIO PARA LA EVALUACIÓN DE COMPETENCIAS ADQUIRIDAS TAL QUE VERIFIQUE EL PERFIL PROFESIONAL”

CASO: CARRERA DE MEDICINA UNIVERSIDAD PUBLICA DE EL ALTO

Para Optar al Título de Licenciatura en Ingeniería de Sistemas

MENCIÓN: INFORMÁTICA Y COMUNICACIONES

Postulante: Univ. Otmar Kevin Aguilar Leon

Tutor Metodológico: Lic. Ing. Dionicio Henry Pacheco Rios

Tutor Revisor: Lic. Roger Navia Gutierrez

Tutor Especialista: M.Sc. Lic. Wendy Yomar Sarmiento Martínez

EL ALTO – BOLIVIA

2023

DECLARACIÓN JURADA DE
AUTENTICIDAD Y RESPONSABILIDAD

Yo, Otmar Kevin Aguilar Leon estudiante con C.I. 9991405 L.P. mediante la presente declaro de manera pública que la propuesta de **TRABAJO DE GRADO** titulada **"SISTEMA WEB DE SEGUIMIENTO INDIVIDUAL UNIVERSITARIO PARA LA EVALUACIÓN DE COMPETENCIAS ADQUIRIDAS TAL QUE VERIFIQUE EL PERFIL PROFESIONAL"** es original, siendo el resultado de mi trabajo personal y no constituye una copia o replica de trabajos similares elaborados.

Autorizo la publicación del resumen de mi propuesta en internet y me comprometo a responder todos los cuestionamientos que se desprenden de su lectura.

Asimismo, me hago responsable ante la universidad o terceros de cualquier irregularidad o daño que pudiera ocasionar por el incumplimiento de los declarado.

De identificarse falsificación, plagio, fraude, o que el TRABAJO DE GRADO haya sido publicado anteriormente asumo las consecuencias y sanciones que de mi acción deriven responsabilizándome por todas las cargas legales que se deriven de ello sometiéndome a las normas establecidas y vigentes de la Carrera de Ingeniería de Sistemas de la Universidad Pública de El Alto.

El Alto, diciembre del 2023

_____

Otmar Kevin Aguilar Leon
9991405 L.P.
ootmar1998@gmail.com

DEDICATORIA

A mis padres, que me han dado la vida y me han educado con amor y cariño. Gracias por estar siempre ahí para mí, por apoyarme en todo lo que hago y por enseñarme los valores que me guían en la vida.

A mis hermanos, que son mis cómplices, mis confidentes y mis mejores amigos. Gracias por estar siempre ahí para mí, por hacerme reír y por compartir conmigo los buenos y los malos momentos.

A mis amigos, que son mi familia elegida. Gracias por estar siempre ahí para mí, por apoyarme en todo lo que hago y por hacerme sentir feliz.

Gracias por ser mi fuente de fortaleza, por creer en mí incluso cuando dudaba de mí mismo, y por ser los verdaderos héroes detrás de cada éxito. Este logro no solo es mío, sino nuestro.

AGRADECIMIENTOS

Quiero expresar mi sincero agradecimiento a la carrera de Ingeniería de Sistemas de la Universidad Pública de El Alto. Este viaje académico ha sido una experiencia transformadora gracias a la excelencia de la educación que he recibido.

Al Lic. Ing. Dionicio Henry Pacheco Rios, mi Tutor Metodológico, agradezco su orientación precisa y sus valiosas sugerencias, que fueron la brújula que guió mi investigación.

Al Lic. Roger Navia Gutierrez, mi Tutor Especialista, agradezco su profundo conocimiento en la materia y su disposición constante para compartir su experiencia, enriqueciendo así mi trabajo.

A la M.Sc. Lic. Wendy Yomar Sarmiento Martínez, mi Tutora Revisora, le agradezco por sus minuciosas revisiones y valiosas observaciones que contribuyeron a la calidad final de este proyecto.

Agradezco de manera especial a la Carrera de Medicina de la Universidad Pública de El Alto, por brindarme la oportunidad de realizar mi Proyecto de Grado, también quiero extender mi agradecimiento a la comisión encargada de hacer el seguimiento, cuya guía y retroalimentación fueron invaluables en cada etapa de este proceso.

ÍNDICE GENERAL

	MARCO PRELIMINAR	0
1.1	INTRODUCCIÓN	1
1.2	ANTECEDENTES	2
1.3	Antecedentes Institucionales	2
1.4	Antecedentes afines	2
1.4.1	Antecedentes internacionales	2
1.4.2	Antecedentes nacionales	3
1.4.3	Antecedentes locales	4
1.5	PLANTEAMIENTO DEL PROBLEMA	5
1.5.1	Problema Principal	5
1.5.2	Problemas secundarios	6
1.5.3	Formulación del problema	6
1.6	OBJETIVOS	6
1.6.1	Objetivo general	6
1.6.2	Objetivos específicos	7
1.7	JUSTIFICACIÓN	7
1.7.1	Justificación Técnica	7
1.7.2	Justificación Económica	8
1.7.3	Justificación Social	8
1.8	METODOLOGÍA	8
1.8.1	Metodología de desarrollo	8
1.8.2	Métricas de calidad al software	9
1.8.3	Estimación de costo	9
1.8.4	Seguridad	10
1.8.5	Pruebas de software	11
1.9	Herramientas	12
1.9.1	Hardware	12
1.9.2	Software	13
1.9.2.1	Nestjs	13
1.9.2.2	MariaDB	14
1.9.2.3	Angular	14
1.9.2.4	PrimeNG	14
1.9.2.5	Git	14

1.10 LIMITES Y ALCANCES 15
1.10.1 Limites 15
1.10.2 Alcances 15
1.11 APORTES 16
MARCO TEÓRICO 17
2.1 Introducción 18
2.2 Sistema 18
2.3 Web 19
2.4 Sistema web 20
2.5 Seguimiento 21
2.6 Evaluación por Competencias 21
2.7 Perfil Profesional 22
2.8 Metodologías de Desarrollo de software 23
2.8.1 Metodologías agiles 23
2.8.2 Metodología XP 25
2.8.2.1 Características de XP 25
2.8.2.2 Principales roles 27
2.8.2.3 Proceso XP 28
2.8.2.4 Fases Metodología XP 28
2.8.2.4.1 Planificación 28
2.8.2.4.2 Diseño 31
2.8.2.4.3 Codificación 32
2.8.2.4.4 Pruebas 33
2.8.2.4.5 Lanzamiento 34
2.9 Herramientas 35
2.9.1 Angular 35
2.9.2 Nestjs 37
2.9.2.1 Características Nestjs 37
2.9.2.2 Conceptos Nestjs 38
2.9.3 Typescript 39
2.9.3.1 Características TypeScript 40
2.9.3.2 Ventajas y Desventajas TypeScript 40
2.9.4 MariaDB 41
2.9.4.1 Historia MariaDB 42

2.9.4.2 Características MariaDB....43
2.9.5 NodeJs....44
2.9.5.1 Funcionamiento NodeJs....45
2.9.5.2 Ventajas NodeJs....46
2.9.6 Git....47
2.9.6.1 Conceptos Basicos de Git....47
2.9.6.2 Ramas....48
2.9.6.3 Ventajas Desarrollo simultáneo....48
2.10 UML....49
2.10.1 Historia UML....50
2.10.2 La finalidad de UML....50
2.10.3 Términos de UML....51
2.10.4 Tipos de diagramas UML....53
2.11 Calidad de Software....55
2.11.1 Norma ISO/IEC 25000....55
2.11.1.1 ISO/IEC 25010....56
2.11.1.2 Adecuación Funcional....56
2.11.1.3 Eficiencia de desempeño....57
2.11.1.4 Usabilidad....57
2.11.1.5 Fiabilidad....58
2.11.1.6 Seguridad....58
2.11.1.7 Mantenibilidad....59
2.11.1.8 Portabilidad....59
2.12 Estimación de costo del desarrollo de Software....60
2.12.1 COCOMO II....61
2.12.1.1 COCOMO II con conteo de líneas de código....62
2.12.1.2 COCOMO II con puntos de función....62
2.13 Seguridad....63
2.13.1 Seguridad del Software....63
2.13.2 Seguridad de la Información....63
2.13.3 ISO 27001:2013....64
2.13.4 Sistema de Gestión de Seguridad de la Información....65
2.13.5 Disponibilidad....66
2.13.6 Confidencialidad....66

2.13.7 Integridad 67
2.14 Pruebas de Software 67
2.14.1 Prueba unitaria 67
2.14.2 Prueba de integración 68
2.14.3 Prueba de aceptación 69
2.14.4 Prueba de caja blanca (estructural, caja transparente) 69
2.14.5 Pruebas de caja negra (funcional, conductual, caja cerrada) 70
2.14.6 Pruebas visuales / de interfaz de usuario (pruebas de navegador) 70
2.14.7 Prueba de estrés 70
2.14.8 Pruebas de inserción 71
MARCO APLICATIVO 72
3.1 Introducción 73
3.1 Fase de planificación 73
3.1.1 Requerimiento Funcionales 73
3.1.2 Requerimiento No Funcionales: 74
3.1.3 Clasificación de roles 74
3.1.4 Historias de usuario 75
3.1.5 Plan de publicación 82
3.1.6 Iteraciones 83
3.1.6.1 Primera Iteración 83
3.1.6.1.1 Desarrollo de tareas de la primera iteración 84
3.1.6.2 Segunda Iteración 87
3.1.6.2.1 Desarrollo de tareas de la Segunda Iteración 89
3.1.6.3 Tercera Iteración 95
3.1.6.3.1 Desarrollo de tareas Tercera Iteración 96
3.1.6.4 Cuarta Iteración 100
3.1.6.4.1 Desarrollo de tareas Cuarte Iteración 101
3.2 Fase de Diseño 104
3.2.1 Diagrama general de casos de uso 104
3.2.2 Diagramas de casos de uso específicos 106
3.2.3 Diagramas de Secuencia 110
3.2.4 Diagrama Entidad Relación 114
3.2.5 Diagrama de Clases 116
3.2.6 Diagrama físico de la base de datos 117

3.2.7 Diseño navegacional de pantallas 118
3.2.8 Diagrama Navegacional 124
3.3 Fase de Codificación 127
3.3.1 Backend 127
3.3.1.1 Modulo app 127
3.3.1.2 Modulo usuario 128
3.3.2 Frontend 131
3.3.2.1 Módulo de inicio de Sesión 131
3.3.3 Interfaz de usuario 135
3.3.3.1 Primera Iteración 135
3.3.3.2 Segunda Iteración 137
3.3.3.3 Tercera iteración 138
3.3.3.4 Cuarta Iteración 142
3.4 Fase de pruebas 146
3.5 Fase de Lanzamiento 160
3.5.1 Despliegue 160
CALIDAD COSTO SEGURIDAD Y PRUEBAS 163
4.1 Introducción 164
4.2 Calidad 164
4.2.1 Análisis de calidad del software 164
4.2.1.1 Funcionalidad 164
4.2.1.2 Fiabilidad 168
4.2.1.3 Usabilidad 170
4.2.1.4 Mantenibilidad 171
4.2.1.5 Portabilidad 172
4.2.1.6 Calidad Global 174
4.3 Costos 175
4.3.1 Costo de elaboración de proyecto 176
4.3.2 Costo total del sistema 177
4.4 Seguridad 177
4.4.1 Acceso controlado 178
4.4.2 Autorización 178
4.4.3 Seguridad física 179
4.4.4 Criptografía 179

4.5 Pruebas.....180
4.5.1 Caja Blanca.....180
4.5.2 Caja Negra.....184
4.5.3 Pruebas de Estrés.....187
CONCLUSIONES Y RECOMENDACIONES.....191
5.1 Conclusiones.....192
5.2 Recomendaciones.....193
BIBLIOGRAFÍA.....194
ANEXOS.....198

ÍNDICE DE TABLAS

Tabla 3.1 *Descripción de roles de usuario en el sistema* 74
Tabla 3.2 *Historia de usuario de inicio de sesión o login* 75
Tabla 3.3 *Historia de usuario Registro Edición Eliminación y Lectura de Estudiantes* 76
Tabla 3.4 *Historia de usuario Registro Edición Eliminación y Lectura de Docentes* 76
Tabla 3.5 *Historia de usuario Registro Edición Eliminación y Lectura de Competencias* 77
Tabla 3.6 *Historia de usuario Registro Edición Eliminación y Lectura de la Asignaturas* 78
Tabla 3.7 *Historia de usuario Registro Edición Eliminación y Lectura de Calificaciones* 78
Tabla 3.8 *Historia de usuario Creación y Registro de Calificaciones de Parciales y Finales* 79
Tabla 3.9 *Historia de usuario Registro de Calificaciones de Practicas* 79
Tabla 3.10 *Historia de usuario Registro de Competencias* 80
Tabla 3.11 *Historia de usuario Lectura de Calificaciones y Competencias* 80
Tabla 3.12 *Historia de usuario Cerrar sesión* 81
Tabla 3.13 *Historia de usuario Inscripción de Estudiantes* 81
Tabla 3.14 *Lista de Historias de usuario* 82
Tabla 3.15 *Tabla de cronograma de publicación de iteraciones* 83
Tabla 3.16 *Historias de usuario de la primera iteración* 83
Tabla 3.17 *Tareas de Primera iteración* 84
Tabla 3.18 *Tarjeta de Tarea n°1* 84
Tabla 3.19 *Tarjeta de Tarea n°2* 85
Tabla 3.20 *Tarjeta de Tarea n°3* 85
Tabla 3.21 *Tarjeta de Tarea n°4* 86
Tabla 3.22 *Tarjeta de Tarea n°5* 86
Tabla 3.23 *Tarjeta de Tarea n°6* 87
Tabla 3.24 *Historias de usuario de la segunda iteración* 87
Tabla 3.25 *Tareas de Segunda iteración* 88

Tabla 3.26 *Tarjeta de Tarea n°7* 89
Tabla 3.27 *Tarjeta de Tarea n°8* 90
Tabla 3.28 *Tarjeta de Tarea n°9* 90
Tabla 3.29 *Tarjeta de Tarea n°10* 90
Tabla 3.30 *Tarjeta de Tarea n°11* 91
Tabla 3.31 *Tarjeta de Tarea n°12* 91
Tabla 3.32 *Tarjeta de Tarea n°13* 92
Tabla 3.33 *Tarjeta de Tarea n°14* 92
Tabla 3.34 *Tarjeta de Tarea n°15* 93
Tabla 3.35 *Tarjeta de Tarea n°16* 93
Tabla 3.36 *Tarjeta de Tarea n°17* 94
Tabla 3.37 *Tarjeta de Tarea n°18* 94
Tabla 3.38 *Tarjeta de Tarea n°19* 94
Tabla 3.39 *Historias de usuario de la tercera iteración* 95
Tabla 3.40 *Tareas de tercera iteración* 95
Tabla 3.41 *Tarjeta de Tarea n°20* 97
Tabla 3.42 *Tarjeta de Tarea n°21* 97
Tabla 3.43 *Tarjeta de Tarea n°22* 97
Tabla 3.44 *Tarjeta de Tarea n°23* 98
Tabla 3.45 *Tarjeta de Tarea n°24* 98
Tabla 3.46 *Tarjeta de Tarea n°25* 99
Tabla 3.47 *Tarjeta de Tarea n°26* 99
Tabla 3.48 *Tarjeta de Tarea n°27* 99
Tabla 3.49 *Tarjeta de Tarea n°28* 100
Tabla 3.50 *Historias de usuario de la cuarta iteración* 100
Tabla 3.51 *Tareas de Cuarta iteración* 101
Tabla 3.52 *Tarjeta de Tarea n°29* 101
Tabla 3.53 *Tarjeta de Tarea n°30* 102
Tabla 3.54 *Tarjeta de Tarea n°31* 102
Tabla 3.55 *Tarjeta de Tarea n°32* 103
Tabla 3.56 *Tarjeta de Tarea n°33* 103

Tabla 3.57 *Tarjeta de Tarea n°34* 104
Tabla 3.58 *Tarjeta de Prueba de aceptación N° 1* 146
Tabla 3.59 *Tarjeta de Prueba de aceptación N° 2* 146
Tabla 3.60 *Tarjeta de Prueba de aceptación N° 3* 147
Tabla 3.61 *Tarjeta de Prueba de aceptación N° 4* 148
Tabla 3.62 *Tarjeta de Prueba de aceptación N° 5* 148
Tabla 3.63 *Tarjeta de Prueba de aceptación N° 6* 149
Tabla 3.64 *Tarjeta de Prueba de aceptación N° 7* 150
Tabla 3.65 *Tarjeta de Prueba de aceptación N° 8* 151
Tabla 3.66 *Tarjeta de Prueba de aceptación N° 9* 151
Tabla 3.67 *Tarjeta de Prueba de aceptación N° 10* 152
Tabla 3.68 *Tarjeta de Prueba de aceptación N° 11* 153
Tabla 3.69 *Tarjeta de Prueba de aceptación N° 12* 153
Tabla 3.70 *Tarjeta de Prueba de aceptación N° 13* 154
Tabla 3.71 *Tarjeta de Prueba de aceptación N° 14* 154
Tabla 3.72 *Tarjeta de Prueba de aceptación N° 15* 155
Tabla 3.73 *Tarjeta de Prueba de aceptación N° 16* 156
Tabla 3.74 *Tarjeta de Prueba de aceptación N° 17* 156
Tabla 3.75 *Tarjeta de Prueba de aceptación N° 18* 157
Tabla 3.76 *Tarjeta de Prueba de aceptación N° 19* 158
Tabla 3.77 *Tarjeta de Prueba de aceptación N° 20* 158
Tabla 3.78 *Tarjeta de Prueba de aceptación N° 21* 159
Tabla 3.79 *Tarjeta de Prueba de aceptación N° 22* 159
Tabla 4.1 *Tabla de obtención de datos de parámetros para el cálculo de fi* 165
Tabla 4.2 *Tabla de ponderación* 166
Tabla 4.3 *Tabla de factor de complejidad* 166
Tabla 4.4 *Tabla para el cálculo de fiabilidad* 169
Tabla 4.5 *Tabla de encuestas sobre usabilidad* 170
Tabla 4.6 *Tabla de cálculo de calidad Global de sistema* 174
Tabla 4.7 *Tabla de costo de elaboración de proyecto* 176
Tabla 4.8 *Tabla de costo total del sistema usando COCOMO* 177

Tabla 4.9 *Tabla de prueba de nodos caso creación de usuario* 181
Tabla 4.10 *Tabla de prueba de nodos caso autenticación de usuario* 183
Tabla 4.11 *Valores caso inicio de sesión* 184
Tabla 4.12 *Prueba de caja negra inicio de sesión* 185
Tabla 4.13 *Campos para creación de usuario* 186
Tabla 4.14 *Prueba de caja negra creación de usuario* 186

ÍNDICE DE FIGURAS

Figura 3.1 *Diagrama de Casos de Uso General* 105
Figura 3.2 *Diagrama de caso de uso: inicio y cierre de sesión* 106
Figura 3.3 *Diagrama de casos de uso: Gestión de usuarios* 106
Figura 3.4 *Diagrama de caso de uso: Administración de parciales y prácticas* 107
Figura 3.5 *Diagrama de caso de uso: Revisar Calificaciones y competencias* 107
Figura 3.6 *Diagrama de caso de uso: inscripción de estudiante* 108
Figura 3.7 *Diagrama de caso de uso: Gestión de competencias* 108
Figura 3.8 *Diagrama de caso de uso: Gestión de asignaturas* 109
Figura 3.9 *Diagrama de secuencia: Administración de usuario* 110
Figura 3.10 *Diagrama de secuencia: Login* 111
Figura 3.11 *Diagrama de secuencia: Inscripción* 111
Figura 3.12 *Diagrama de secuencia: Administración de calificaciones y competencias* 112
Figura 3.13 *Diagrama de secuencia: Administración de asignaturas* 113
Figura 3.14 *Diagrama de secuencia: Administración de Competencias* 114
Figura 3.15 *Diagrama Entidad Relación General* 115
Figura 3.16 *Diagrama de clases del sistema* 116
Figura 3.17 *Diagrama físico de la base de datos* 117
Figura 3.18 *Login de inicio de sesión* 118
Figura 3.19 *Pantalla de inicio con los módulos accesibles por el rol* 119
Figura 3.20 *Módulo de estudiantes* 119
Figura 3.21 *Módulo de materias* 120
Figura 3.22 *Módulo de docentes* 120
Figura 3.23 *Módulo de competencias* 121
Figura 3.24 *Módulo de inscripciones* 121
Figura 3.25 *Login e Inicio en diseño responsivo* 122
Figura 3.26 *Materias de docente y Administración de materia* 122
Figura 3.27 *Módulo de detalle de estudiante* 123
Figura 3.28 *Diagrama navegacional de Docentes* 124
Figura 3.29 *Diagrama navegacional estudiantes* 125

Figura 3.30 Diagrama navegacional administrado 126
Figura 3.31 *Vista Interfaz de login* 135
Figura 3.32 *Vista Panel de administración* 136
Figura 3.33 *Vista Panel de administración de estudiantes* 136
Figura 3.34 *Vista Panel de administración de docentes* 137
Figura 3.35 *Vista Panel de administración de asignaturas* 137
Figura 3.36 *Vista Panel de administración de competencias* 138
Figura 3.37 *Listado de estudiantes* 138
Figura 3.38 *Vista Panel de administración de Parciales* 139
Figura 3.39 *Vista Panel de administración de practicas* 139
Figura 3.40 *Panel de asignaturas de docente* 140
Figura 3.41 *Calificar estudiante* 140
Figura 3.42 *Dialogo de introducción de calificación* 141
Figura 3.43 *Asignación de competencia a estudiante* 141
Figura 3.44 *Vista Inicio del sistema estudiante* 142
Figura 3.45 *Vista Inicio del sistema estudiante responsive* 142
Figura 3.46 *Perfil usuario* 143
Figura 3.47 *Calificaciones de estudiante (vista responsive)* 143
Figura 3.48 *Competencia de estudiante (vista responsive)* 144
Figura 3.49 *Asignaturas de estudiante (vista responsive)* 144
Figura 3.50 *Módulo de inscripciones* 145
Figura 3.51 *Inscripción de estudiante* 145
Figura 3.52 *Configuración de conexión Railway backend con base de datos* 161
Figura 3.53 *Sitio de alojamiento de la aplicación* 162
Figura 4.1 *Costos con la página web COCOMO II* 175
Figura 4.2 *Resultados COCOMO II* 176
Figura 4.3 *Fragmento de código creación de usuarios modulo de usuarios* 180
Figura 4.4 *Grafo del código de Creación de usuario* 181
Figura 4.5 *Fragmento de código modulo autenticación de usuarios* 182
Figura 4.6 *Grafo del código de autenticación de usuario* 183
Figura 4.7 *Login de sistema* 184

Figura 4.8 *Formulario de creación de usuario estudiante* 185
Figura 4.9 *Solicitud al servidor endpoint creación de usuario* 188
Figura 4.10 *Respuesta la prueba del servidor al endpoint de creación de usuarios* 188
Figura 4.11 *Grafico de respuesta con JMeter* 189
Figura 4.12 *Respuestas a solicitud con JMeter* 189
Figura 4.13 *Sumario de Respuestas con JMeter* 190

RESUMEN

Los sistemas de información y las tecnologías de la información han transformado la forma en que operan las instituciones en la actualidad. Su uso permite mejorar el manejo de la información, automatizando procesos administrativos de forma más eficiente y ágil.

La carrera de Medicina de la Universidad Pública de El Alto se encuentra en un proceso de acreditación internacional. En este contexto, las nuevas tecnologías son un pilar fundamental para el tratamiento de la información. Por ello, se requiere una herramienta que facilite el control de las competencias de los estudiantes universitarios, las cuales se adquieren a lo largo de la carrera. Esta herramienta será de gran ayuda para verificar el perfil profesional de los estudiantes, requisito para la toma de decisiones.

Se establecen tres tipos de competencias que los estudiantes deben adquirir: las macrocompetencias son habilidades y capacidades que no dependen de una disciplina en particular, como la capacidad de trabajar en equipo o la capacidad de resolver problemas, las competencias genéricas son habilidades y capacidades que todos los profesionales deben tener, como la capacidad de comunicarse de manera efectiva o la capacidad de trabajar de manera autónoma y las competencias específicas son habilidades y capacidades que los estudiantes de medicina deben tener al finalizar sus estudios, como la capacidad de diagnosticar enfermedades o la capacidad de realizar intervenciones quirúrgicas.

El proyecto actual consiste en diseñar y desarrollar un sistema que permita el seguimiento individual de los conocimientos adquiridos por cada estudiante universitario. Estos datos serán utilizados para valorar el perfil profesional de los estudiantes. El sistema se desarrollará bajo la metodología XP, que garantiza una buena organización del flujo de trabajo y el cumplimiento de los plazos de entrega. Para el desarrollo se utilizarán las siguientes tecnologías: Angular, NestJs, MariaDB, TypeScript, NodeJs y Git.

ABSTRACT

Information systems and information technologies have transformed the way institutions operate today. Their use allows for the improvement of information management by automating administrative processes more efficiently and quickly.

The Medicine program at the Public University of El Alto is undergoing an international accreditation process. In this context, new technologies are a fundamental pillar for handling information. Therefore, a tool is needed to facilitate the control of university students' competencies acquired throughout their academic journey. This tool will be of great help in verifying the professional profile of students, a requirement for decision-making.

Three types of competencies that students must acquire are established: macro-competencies are skills and abilities that do not depend on a particular discipline, such as the ability to work in a team or problem-solving skills. Generic competencies are skills that all professionals should have, like effective communication or the ability to work autonomously. Specific competencies are skills that medicine students must have upon completing their studies, such as the ability to diagnose diseases or perform surgical interventions.

The current project involves designing and developing a system that allows for the individual tracking of knowledge acquired by each university student. These data will be used to assess the professional profile of students. The system will be developed using the XP methodology, ensuring a good organization of the workflow and meeting delivery deadlines. The following technologies will be used for development: Angular, NestJs, MariaDB, TypeScript, NodeJs, and Git.

LISTADO DE ABREVIATURAS

XP	Extreme programming
COCOMO	Constructive cost model
JWT	Json web token
HMAC	Hash-based message authentication code
HTML	Hyper text markup language
HTTP	Hypertext transfer protocol
MDN	Mozilla developer network
MS	Mantenlo sencillo
CPU	Central processing unit
OOP	Object orientes programming
FR	Funtional programming
FRP	Function Reactive programming
POO	Programación orientada a objetos
API	Interfaz de programación de aplicaciones
HOF	High order functions
DB	Database
SQL	Lenguaje de consulta estructurada
PAM	Pluggable *authentication* modules
LDAD	Lightweight directory access protocol
PHP	Hypertext pre-processor
IDE	Entorno de desarrollo integrado
UML	Lenguaje unificado modulado
MOF	Meta-object facility
DML	Metamodelo de Almacén Común
LM	Construcciones de metamodelos
MDA	Arquitectura dirigida por modelos
OCL	Lenguaje de restricciones de objetos
OMG	Objet management group
SQuaRE	Sistem and software quality requiremets and evaluation
ISO	Organización Internacional de Normalización
SGSI	Sistema de seguridad de la Informacion
DDOS	Ataque distribuido de denegación de servicio
HA	High availability

MARCO PRELIMINAR

1.1 INTRODUCCIÓN

Los sistemas de información y las tecnologías de información han cambiado la forma en la que operan las instituciones actualmente. A través de su uso se logran importantes mejoras en el manejo de la información, ya que automatizan procesos administrativos de una manera más eficaz y ágil.

La acreditación es el proceso que apunta a garantizar la excelencia de una institución o de un programa educativo, ya que supone una evaluación basada en estándares y criterios de calidad establecidos previamente por una agencia u organismo acreditador, por lo tanto la carrera de Medicina de la Universidad Pública de El Alto, en ese sentido entiende que las nuevas tecnologías son un pilar importante para el tratamiento de la información, por lo que se requiere una herramienta que coadyuve a facilitar el control de las calificaciones y competencias de cada estudiante universitario el cual acumula a lo largo de su carrera, la misma será de gran ayuda para verificar el perfil profesional, requisito para la toma de decisiones dentro de la institución.

El actual proyecto consiste en diseñar y desarrollar un sistema, que permita el seguimiento individual de los conocimientos adquiridos de cada estudiante universitario, datos que coadyuvaran para a la valoración del perfil profesional, dicho sistema estará realizado bajo la metodología XP la cual nos permitirá una buena organización del flujo de trabajo, también cumplimiento con los tiempo de entrega, se aplicara las siguientes tecnologías para el desarrollo: Angular, NestJs, MariaDB, TypeScript, NodeJs y Git.

1.2 ANTECEDENTES

1.3 Antecedentes Institucionales

La población de la ciudad de El Alto aproximadamente desde el año 1998 gestiona el contar con una Universidad pública en el afán de llenar una muy sentida necesidad en lo que se refiere a la educación superior, es así que luego de una ardua lucha para lograr este anhelo, el 1 de Mayo del 2000, una marcha de más de 25000 personas entre estudiantes, profesionales y organizaciones sociales diversas, toman los predios de Villa Esperanza que pertenecían a la U.M.S.A(Universidad Mayor de San Andrés). declarándolos pertenencia de la futura Universidad Pública de El Alto.

La carrera de Medicina tiene la siguiente misión y visión:

Visión: Formar profesionales de la salud, médicos generales, críticos y humanistas, con valores éticos, compromiso social y voluntad de superación, que responden a las necesidades de salud con innovación y excelencia educativa, capases de difundir conocimientos, tecnología y avances científicos, para contribuir a la salud y desarrollarse en la investigación científica en beneficio del ser humano y del país.

Misión: Formar profesionales de la salud, médicos generales, críticos y humanistas, con valores éticos, compromiso social y voluntad de superación, que responden a las necesidades de salud con innovación y excelencia educativa, capases de difundir conocimientos, tecnología y avances científicos, para contribuir a la salud y desarrollarse en la investigación científica en beneficio del ser humano y del país.

1.4 Antecedentes afines

1.4.1 Antecedentes internacionales

- Leyva, A. L. (2021). Sistema Multiplataforma para el Acompañamiento y Seguimiento de las competencias en Estudiantes de Instituciones Educativas Básicas Públicas. Tesis para obtener el Título Profesional de Ingeniero de Sistemas. Universidad Peruana Unión, Tarapoto, Perú.

El proyecto habla sobre cómo se debe proporcionar espacios para que un estudiante desarrolle su actividad académica de manera adecuada para su preparación. Este

artículo se focaliza en determinar la mejora de la gestión del proceso de acompañamiento y seguimiento de esas competencias en los estudiantes de educación secundaria de una institución pública en Amazonas –Perú.

- Ricardo, B. C. (2020). Sistema informático transaccional orientado a la web para el control académico en el instituto labriega mistral de la ciudad de santo domingo. Proyecto de investigación previo a la obtención del título de ingeniero en sistemas e informática. Universidad Regional Autónoma De Los Andes, Santo Domingo, Ecuador.

La problemática de este trabajo de grado interactúa con los diferentes roles que se encuentra en la Institución para el control académico cuyo enfoque tiene la inscripción de los estudiantes, los registros de las calificaciones y los pagos realizados.

- Acevedo Quispe, Y. L. (2018). Implementación de un sistema web para la mejora del proceso administrativo académico de la institución educativa “Wari-Vilca”- Huayucachi. Para optar el título profesional de ingeniera de sistemas. Universidad nacional del centro del Perú, Huancayo, Perú.

Indagan sobre las dificultades que hay en el proceso de registro de la información de los estudiantes y el cómo la implementación de un sistema web si lograría un mejor manejo de la información.

1.4.2 Antecedentes nacionales

- Alarcon Aroyo, O. (2017). Sistema web para el control y seguimiento de kárdex administrativo Caso: postgrado en Informática. Para optar al título de licenciatura en informática. Universidad Mayor De San Andrés, La paz, Bolivia.

Se desarrollo un sistema para el Kárdex Administrativo de la Unidad del Postgrado en Informática, el cual está encargado de manejar de manera correcta y eficaz la información que hace su entrante y saliente del Postgrado en Informática, acerca de los participantes de los programas de la Unidad, los programas en curso, los datos de los docentes, las notas respectivas de cada participante al igual que sus pagos correspondientes por sus programas.

- Rodríguez Poma, H. E. (2021). Sistema de encuesta web para el seguimiento a los titulados de la Universidad Mayor de San Andrés. Proyecto de Grado para obtener el Título de Licenciatura en Informática. Universidad Mayor de San Andrés, La paz, Bolivia.

Realiza encuestas a estudiantes titulados de cada carrera para conocer su opinión acerca de su formación, es una encuesta hecha por los encargados de cada carrera, y para el almacenamiento de los datos se hará uso del sistema de gestión de base de datos PostgreSQL para que al final se pueda realizar los cuadros estadísticos.

- Dorado Gómez, M. A. (2019). Plataforma GITLAB como Estrategia de Aprendizaje Colaborativo en el rendimiento académico de los estudiantes de la Carrera de Sistemas Informáticos del I.T.M.Q.S.C. Sede Central. Tesis de Maestría para optar el grado académico de magister scientiarum en educación superior. Universidad Mayor de San Andrés, La Paz, Bolivia.

Se realiza una investigación de enfoque cuantitativo, cuasi experimental de tipo descriptivo correlacional, Que está orientado a describir la incidencia de la implementación de la GitLab como estrategia de aprendizaje colaborativo.

1.4.3 Antecedentes locales

- Calamani Salas, J. M. (2020). “Sistema de información para el registro y seguimiento académico” caso: academia de formación profesional "Bela". Para optar al título de Licenciatura en Ingeniería de Sistemas. Universidad Pública De El Alto, El Alto, Bolivia.

Señala que luego de la implantación del proyecto contribuyo en el flujo de las actividades administrativas así mismo entrega información oportuna clara y concisa.

- Tola Mendoza, M. (2020). “Sistema de información web para el control y seguimiento académico” caso: Instituto técnico San Pablo. Para Optar al Título de Licenciatura en Ingeniería de Sistemas. Universidad Pública del Alto, El Alto, Bolivia.

El proyecto Habla del manejo de la información en instituciones de carácter educativo no son una excepción ya que se debe prestar atención al uso de los datos y la información, señala que antes de la implementación del sistema se generaban retrasos y con el sistema aquellos problemas se solucionaron.

- Villarroel Apaza, V. (2020). Sistema web para la gestión de procesos de pasantías y prácticas profesionales. Para optar al título de Licenciatura en Ingeniería de Sistemas. Universidad Pública de el Alto, El Alto, Bolivia.

Indica que el proceso de pasantía se la realiza de manera manual, es decir a través de documentación de manera física, lo que provoca inconvenientes en cuanto al seguimiento del desarrollo de las pasantías tales como retraso en la disponibilidad de información, mala organización de los documentos reglamentarios, pérdida de evidencias, incumplimiento de tareas programadas

1.5 PLANTEAMIENTO DEL PROBLEMA

1.5.1 Problema Principal

La carrera de Medicina de la Universidad Pública De El Alto no tiene las herramientas para realizar un correcto seguimiento a los estudiantes universitarios sobre su aprendizaje y así poder verificar su perfil profesional en función a las competencias que debe tener cada estudiante universitario, lo cual conlleva a no tener información oportuna del mismo en su proceso de formación profesional.

La carrera de medicina cuenta con las siguientes competencias:

- Macro competencias: Son habilidades, capacidades que no dependen de una disciplina en particular.
- Competencias Genéricas: Son las capacidades que todos los profesionales deben tener.
- Competencias Especificas: son capacidades que los universitarios deben tener al finalizar los estudios de medicina.

Estas competencias son datos cualitativos y deben ser adquiridos por los universitarios como requisito indispensable para completar con su perfil profesional, por otra parte,

es importante para la toma de decisiones con el objetivo de mejorar el sistema de aprendizaje.

Así mismo se tiene la necesidad de centralizar las calificaciones lo cuales son datos cuantitativos. obtenidos durante el año académico como resultado de los exámenes parciales, finales y prácticas, para obtener seguimiento académico de los universitarios.

1.5.2 Problemas secundarios

- No se tiene los datos de las competencias de forma accesible por lo que no se puede validar el proceso de cumplimiento del plan de estudios de cada universitario.
- No se tiene el detalle centralizado de las calificaciones de los exámenes parciales, finales y prácticas lo que ocasiona que no se cuente con información actualizada y consolidada para el seguimiento de los universitarios.
- Demora en la elaboración de informes, ya que no se cuenta con los datos de forma oportuna.
- La generación de información no es rápida lo que ocasiona retraso en los tramites de los universitarios.

1.5.3 Formulación del problema

¿De qué manera se podría agilizar la gestión y el seguimiento de la información sobre las competencias y calificaciones que adquiere el estudiante universitario, para la validación de su perfil profesional de una manera confiable, oportuna y segura?

1.6 OBJETIVOS

1.6.1 Objetivo general

Desarrollar e implementar un “Sistema web de seguimiento individual universitario para la evaluación de competencias adquiridas tal que verifique el perfil profesional”, el cual proporcione información oportuna respecto de las competencias adquiridas y calificaciones del estudiante universitario de una manera confiable y segura, para la

óptima toma de decisiones, en la carrera de medicina de la Universidad Pública de El Alto.

1.6.2 Objetivos específicos

- Realizar el análisis de la situación actual sobre gestión de la información de las calificaciones y competencias.
- Sistematizar el proceso de generación de información del estudiante universitario sobre sus competencias y calificaciones adquiridas de manera clara y concisa.
- Diseñar una base de datos la cual haga un manejo adecuado de la información a largo plazo y de forma segura.
- Crear una interfaz sencilla e intuitiva, cuidando la accesibilidad de los usuarios tanto para docentes, estudiantes y administradores.
- Brindar Información oportuna docentes y administrativos sobre las competencias adquiridas del estudiante universitario para la verificación de su perfil profesional.

1.7 JUSTIFICACIÓN

1.7.1 Justificación Técnica

La Carrera de Medicina de la Universidad Pública de El Alto cuenta con los equipos necesarios para acceder al sistema a desarrollar ya que al ser un sistema web se podrá acceder desde cualquier dispositivo que pueda navegar por internet ya sea computadoras de escritorio, tablets y teléfonos móviles.

Por la parte de despliegue de la aplicación la Universidad Pública de El Alto cuenta con el departamento SIE (Sistemas de Información y Estadística) el cual brindara la infraestructura necesaria para desplegar el sistema web.

1.7.2 Justificación Económica

El proyecto una vez desarrollado e implementado logrará la disminución en el trabajo del personal que se involucra en el trámite administrativo para la verificación del perfil profesional.

El sistema será desarrollado con software de código abierto el cual no emana gasto alguno por lo que no existirá un costo en las herramientas utilizadas en el desarrollo, así mismo al ser un sistema web habrá una disminución el uso del papel e insumos de oficina.

1.7.3 Justificación Social

El sistema tiene como usuarios directos a personal administrativo, docentes y a la población universitaria de la carrera de Medicina de la Universidad pública de el Alto, podrán obtener información de manera oportuna e integra sobre las competencias adquiridas y calificaciones de los estudiantes universitarios para la verificación del perfil profesional.

Por otra parte, es importante para los usuarios indirectos como las autoridades de la carrera y de la Universidad pública de el Alto, siendo que se tendrá la información de forma digital y almacenada para responder a cualquier solicitud posterior.

1.8 METODOLOGÍA

1.8.1 Metodología de desarrollo

Analizando las características del proyecto a desarrollar se toma como marco de desarrollo las metodologías agiles.

> “Por definición, las metodologías ágiles son aquellas que permiten adaptar la forma de trabajo a las condiciones del proyecto, consiguiendo flexibilidad e inmediatez en la respuesta para amoldar el proyecto y su desarrollo a las circunstancias específicas del entorno” (Sotomayor, 2021).

En ese entendido se eligió la metodología XP,

> La metodología XP es un conjunto de técnicas que dan agilidad y flexibilidad en la gestión de proyectos. También es conocida como Programación Extrema (Extreme Programming) y se centra crear un producto según los requisitos exactos del cliente. De ahí, que le involucre al máximo durante el método de gestión del desarrollo del producto. (SINNAPS, 2020)

Entre las características más resaltantes de la metodología XP se tiene:

- Comunicación constante entre el cliente y el equipo de desarrollo.
- Respuesta rápida a los cambios constantes.
- La planificación es abierta con un cronograma de actividades flexible.
- El software que funciona está por encima de cualquier otra documentación.
- Los requisitos del cliente y el trabajo del equipo del proyecto son los principales factores de éxito del mismo.

1.8.2 Métricas de calidad al software

El presente proyecto está en el margen del estándar ISO/IEC 25010 la cual presenta las siguientes características métricas de calidad: Funcionalidad, Confiabilidad, Usabilidad, Eficiencia, Capacidad de mantenimiento y Portabilidad, al hacer la aplicación de estas características se asegurarán que el software cumpla con todas las métricas de calidad necesarias.

1.8.3 Estimación de costo

Una de las tareas de mayor importancia en la planificación de proyectos de software es la estimación, la cual consiste en determinar, con cierto grado de certeza, los recursos de hardware y software, costo, tiempo y esfuerzo necesarios para el desarrollo de los mismos.

En ese entendido se usará COCOMO II, este modelo permite realizar estimaciones en función del tamaño del software, y de un conjunto de factores de costo y de escala, se engloba en el grupo de los modelos algorítmicos que tratan de establecer una relación

matemática la cual permite estimar el esfuerzo y tiempo requerido para desarrollar un producto.

1.8.4 Seguridad

Por la parte de seguridad se tomará en cuenta los siguientes controles que se especifican dentro del marco de seguridad ISO 27001:2013:

- Acceso controlado
- Autorización
- Seguridad Física
- Criptografía

Los requisitos de la Norma ISO 27001:2013 norma nos aportan un Sistema de Gestión de la Seguridad de la Información (SGSI), consistente en medidas orientadas a proteger la información, indistintamente del formato de la misma, contra cualquier amenaza, de forma que garanticemos en todo momento la continuidad de las actividades de la empresa. (Normas ISO, 2017).

Tal norma indica que como objetivo a cumplir las siguientes características:

- Confidencialidad
- Integridad
- Disponibilidad

> Como una de los más resaltantes beneficios es “Los riesgos de seguridad de la información representan una amenaza considerable para las empresas debido a la posibilidad de pérdida financiera o daño, la pérdida de los servicios esenciales de red, o de la reputación y confianza de los clientes”. (Normas ISO, 2017)

Así también los frameworks con la cuales se desarrolló el sistema nos brindan de ciertas herramientas que nos ayudan a cumplir con ciertos parámetros de seguridad.

A nivel de seguridad del sistema se utilizará JWT tanto para la autenticación como la protección de rutas.

JSON Web Token (JWT) is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. This information can be verified and trusted because it is digitally signed. JWTs can be signed using a secret (with the HMAC algorithm) or a public/private key pair using RSA or ECDSA [JSON Web Token (JWT) es un estándar abierto (RFC 7519) que define una forma compacta y autónoma de transmitir información de forma segura entre las partes como un objeto JSON. Esta información se puede verificar y confiar porque está firmada digitalmente. Los JWT se pueden firmar usando un secreto (con el algoritmo HMAC) o un par de claves pública/privada usando RSA o ECDSA] (Jwt.io, 2013)

La seguridad en la base de datos esta cuidada mediante el uso de las buenas prácticas las cuales son:

- Controles de acceso administrativo y de red: el número mínimo práctico de usuarios debe tener acceso a la base de datos, y sus permisos se deben limitar a los niveles mínimos necesarios para que puedan realizar su trabajo. Asimismo, el acceso a la red debe limitarse al nivel mínimo de permisos necesarios.
- Seguridad de software de base de datos: utilice siempre la última versión del software de gestión de bases de datos y aplique todos los parches en cuanto se publiquen.
- Seguridad del servidor web/de aplicaciones: cualquier aplicación o servidor web que interactúe con la base de datos puede ser un canal para el ataque y debe estar sujeto a pruebas de seguridad constantes y contar con prácticas recomendadas de gestión. (Education IBM Cloud, 2019)

1.8.5 Pruebas de software

Para poner a prueba el software se tomó algunas de las pruebas más relevantes en función al tipo de software a realizarse la cuales son:

Pruebas de integración.

Las pruebas de integración verifican que los distintos módulos o servicios utilizados por tu aplicación funcionan bien en conjunto. Por ejemplo, se puede probar la interacción con la base de datos o asegurarse de que los microservicios funcionan bien en conjunto y según lo esperado. Estos tipos de pruebas son más costosos de ejecutar, ya que requieren que varias partes de la aplicación estén en marcha.

Pruebas funcionales.

Las pruebas funcionales se centran en los requisitos empresariales de una aplicación. Solo verifican el resultado de una acción y no comprueban los estados intermedios del sistema al realizar dicha acción.

Pruebas de rendimiento.

Las pruebas de rendimiento comprueban el comportamiento del sistema cuando está sometido a una carga importante. Estas pruebas no son funcionales y pueden tener la forma variada de entender la fiabilidad, la estabilidad y la disponibilidad de la plataforma; por ejemplo, puede estar observando tiempos de respuesta al ejecutar un gran número de solicitudes, o cómo se comporta el sistema con una cantidad significativa de datos (Atlassian, 2022).

1.9 Herramientas

1.9.1 Hardware

Una de las herramientas necesarias al momento de realizar desarrollo web es una computadora la cual debería cumplir con las siguientes características:

- Procesador: Intel i5 o AMD Ryzen 5 en adelante.
- Memoria RAM: 16 GB.
- Almacenamiento: disco SSD (estado sólido).
- Conexión a internet.

Sin embargo, estos requisitos son flexibles.

Para la implementación del proyecto deberá contar con el hardware recomendable de:

- 2 Cores 3.2 GHz. o superior
- 2 Gb de RAM.
- 16 Gb de almacenamiento.

Para el uso del software desarrollado los equipos o dispositivos deberán cumplir con las siguientes características:

- Procesador: Dual Core o superior
- Memoria RAM: 2 Gb o superior
- Almacenamiento: 120 GB o mas
- Sistema Operativo: Windows, Linux, Mac
- Otros: Conexión a internet

1.9.2 Software

Para el desarrollo de la plataforma del lado del Backend se empleó las siguientes herramientas:

1.9.2.1 Nestjs

Nest (NestJS) is a framework for building efficient, scalable Node.js server-side applications. It uses progressive JavaScript, is built with and fully supports TypeScript (yet still enables developers to code in pure JavaScript) and combines elements of OOP (Object Oriented Programming), FP (Functional Programming), and FRP (Functional Reactive Programming) [Nest (NestJS) es un marco para crear aplicaciones del lado del servidor Node.js eficientes y escalables. Utiliza JavaScript progresivo, está construido con TypeScript y es totalmente compatible (pero aún permite a los desarrolladores codificar en JavaScript puro) y combina elementos de OOP (Programación orientada a objetos), FP (Programación funcional) y FRP (Programación reactiva funcional)], (NestJS, 2022).

1.9.2.2 MariaDB

MariaDB Server es un sistema de gestión de bases de datos relacionales de código abierto. Es uno de los servidores de bases de datos más populares del mundo, con usuarios notables como Wikipedia, WordPress.com y Google. MariaDB Server se publica bajo la licencia de código abierto GPLv2 y se garantiza entonces que seguirá siendo abierto. (MariaDB Foundation, 2022)

En la parte de frontend será desarrollado con las siguientes tecnologías:

1.9.2.3 Angular

El Frontend se empleó Angular en cual en sitio oficial señala: “Angular is an application design framework and development platform for creating efficient and sophisticated single-page apps” [Angular es un marco de diseño de aplicaciones y una plataforma de desarrollo para crear aplicaciones eficientes y sofisticadas de una sola página]. (Angular, s.f.)

1.9.2.4 PrimeNG

PrimeNG es una colección de componentes de interfaz de usuario para Angular. Todos los widgets son de código abierto y de uso gratuito bajo la licencia MIT. PrimeNG es desarrollado por PrimeTek Informatics, un proveedor con años de experiencia en el desarrollo de soluciones UI de código abierto.

Otras Herramientas:

1.9.2.5 Git

Para mantener un control de versiones se utilizará Git el cual nos permitirá ir desarrollando las características de la plataforma de una manera más controlable y escalable.

1.10 LIMITES Y ALCANCES

1.10.1 Limites

El Sistema de seguimiento individual de los estudiantes universitarios de su aprendizaje para la verificación de su perfil profesional contempla las siguientes limitaciones:

- El proyecto se limita a proporcionar información acerca del seguimiento de su aprendizaje individual de cada estudiante universitario para lograr las competencias y calificaciones que requiere la carrera de medicina por lo que no se podrá usar como una plataforma educativa.
- El sistema está diseñado para el uso de estudiantes, docentes y personal administrativo.
- No registrará la asistencia a clases de los universitarios.

1.10.2 Alcances

Los alcances del sistema de manera general contemplan los siguientes módulos:

- Registro de estudiantes y docentes: El sistema realiza el registro de estudiantes nuevos y antiguos con todos los datos personales que solicita la institución. El sistema realiza el registro de los docentes con sus datos personales, ellos realizan control y registro de las notas, prácticas y competencias.
- Registro de competencias: El registro de las competencias las cual son las características que deben poseer los estudiantes.
- Registro de Asignaturas: El registro de las asignaturas las cuales se desarrollan dentro de la malla curricular de la carrera de Medicina.
- Módulo de Inscripción: Se lleva el control de las inscripciones de cada una de las asignaturas del estudiante.
- Módulo de gestión de asignatura: El docente lleva el control de las calificaciones de los parciales y de las prácticas de la asignatura la cual desarrolla, en dicho módulo de igual manera se califica y se da las competencias que el estudiante adquiere.

- Módulo de calificaciones y competencias para estudiante: El estudiante puede ingresar a consultar sus calificaciones de las asignaturas en las cuales está inscrito.

1.11 APORTES

Dentro de los aportes del presente proyecto se tienen:

- Lograr consolidar de forma digital los datos de seguimientos del estudiante sobre sus calificaciones y competencias adquiridas
- Facilitar el acceso a la información para la verificación del perfil profesional.

MARCO TEÓRICO

2.1 Introducción

En este capítulo se introducirán los conceptos más relevantes sobre las metodologías, métodos y herramientas utilizadas para el desarrollo del presente proyecto de grado.

2.2 Sistema

Un sistema se define como un conjunto organizado de componentes interrelacionados, ya sean materiales o conceptuales, que posee una estructura, una composición y un entorno específicos. Este término abarca diversas áreas del conocimiento, como la física, la biología, la informática y la computación. La perspectiva sistemática o sistematicista permite abordar el mundo, considerando que todos los objetos forman parte de algún tipo de sistema, desde las partículas de un átomo hasta la corteza cerebral, la democracia representativa o los números enteros. Desde esta óptica, un sistema se percibe como un segmento de la realidad que puede ser estudiado de forma independiente, aunque sus componentes están interconectados. (Etecé, 2021)

> Un sistema consiste en un conjunto ordenado de reglas, principios y procedimientos relacionados entre sí para funcionar orgánicamente. Este conjunto sistemático puede regular el funcionamiento de una cosa, de un grupo o colectividad. Así podemos hablar de un sistema político, un sistema gramatical, un sistema informático, entre otros. La palabra sistema deriva del término systema y σύστημα del latín y griego respectivamente. El significado original de esta palabra hace referencia a un conjunto, agregado o reunión de cosas. Se trata de un objeto complejo cuyas partes se relacionan entre sí, un sistema puede ser de tipo conceptual o material. Los sistemas se caracterizan por contar con cierta composición, entorno y estructura. Aquellos sistemas calificados como materiales cuentan además con mecanismos o procesos. Y entre los sistemas materiales, solo una parte cuenta con figura o forma delimitada. Aunque la corriente filosófica denominada sistemismo considera que todos los objetos son sistemas o partes de un sistema. (Tilio, 2023, párr. 1)

Un sistema se configura como una entidad integral en la cual cada componente establece conexiones con otros, ya sea de manera material o conceptual. Mientras que los métodos exhiben composición, organización y alcance, es exclusivo de los sistemas de naturaleza material incorporar mecanismos, y solo algunos presentan una figura o configuración. Se identifican diversos tipos de sistemas, como los abstractos, físicos, concretos, abiertos o cerrados, algunos de los cuales se clasifican según su composición o su naturaleza. (Martinez, 2022, párr. 1)

2.3 Web

El vocablo "Web" proviene del inglés y significa red o telaraña. También conocido como World Wide Web (www) o simplemente "la Web", hace referencia al sistema de documentos o páginas web interconectados mediante enlaces de hipertexto disponibles en Internet. Inicialmente, el primer navegador web creado adoptó el nombre de Nexus en una etapa posterior.

En la actualidad, esta palabra es de uso frecuente, y ciertos términos han arraigado en el lenguaje cotidiano, como el caso de "sitio web" para referirse al conjunto de páginas web, generalmente asociadas a un dominio o subdominio. Asimismo, se menciona comúnmente el "servidor web" para designar al programa encargado de transferir hipertextos, páginas web o páginas HTML mediante el protocolo HTTP.

Desde el año 2004, ha surgido una nueva expresión en el ámbito de Internet conocida como Web 2.0. Esta noción se relaciona con la segunda generación de la web, que se centra en fomentar la colaboración y el intercambio de información entre usuarios a través de comunidades como redes sociales, blogs, wikis, entre otros. (Cristian, 2018)

> En MDN (2022) nos indica:
>
> La World Wide Web comúnmente conocida como WWW, W3, o la Web es un sistema interconectado de páginas web públicas accesibles a través de Internet. La Web no es lo mismo que el Internet: la Web es una de las muchas aplicaciones construidas sobre Internet.

> Tim Berners-Lee propuso la arquitectura de lo que es conocido como la World Wide Web. Él creó el primer servidor web (server (en-US)), el primer navegador de internet (browser), y la primera página web, en su computadora del laboratorio de investigación de física del CERN en 1990. En 1991, anunció su creación en el grupo de noticias alt.hypertext, marcando con esto el momento en que la Web se hizo pública

Por otra parte, en Techopedia (2022) señala

> La Web es el nombre común de la World Wide Web, un subconjunto de Internet que consiste en las páginas a las que se puede acceder mediante un navegador Web. Muchas personas asumen que la Web es lo mismo que Internet y usan estos términos indistintamente. Sin embargo, el término Internet en realidad se refiere a la red global de servidores que hace posible el intercambio de información que ocurre en la Web. Entonces, aunque la Web constituye una gran parte de Internet, no son lo mismo

2.4 Sistema web

Se denomina sistema web a aquellas aplicaciones de software que puede utilizarse accediendo a un servidor web a través de Internet o de una intranet mediante un navegador, las aplicaciones web son muy usadas hoy en día, debido a lo práctico del navegador web como cliente ligero, a la independencia del sistema operativo. (geefe.com, s.f, párr.1)

Los sistemas web tienen la peculiaridad de estar alojados en un servidor de internet o sobre una intranet (red local/privada), lo cual hace que ellos no dependan de ser instalados sobre una plataforma o sistema operativo en específico. Su aspecto es muy similar a un sitio web que vemos normalmente, pero en realidad los sistemas web van un paso más allá, porque cuentan con funcionalidades muy potentes que brindan respuesta a casos muy particulares.

Cabe recalcar que estos sistemas son totalmente compatibles en cualquier navegador web (Google Chrome, Firefox, Edge, Safari, etc.), este tipo de diferencia se ve reflejada

en los costos, en la rapidez de obtención de la información, en la optimización de tareas y por alcanzar una gestión estable. (ATURA, 2021, párr. 1)

Los «sistemas Web» o también conocido como «aplicaciones Web» son aquellos que están creados e instalados no sobre una plataforma o sistemas operativos (Windows, Linux). Sino que se alojan en un servidor en Internet o sobre una intranet (red local). Su aspecto es muy similar a páginas Web que vemos normalmente, pero en realidad los 'sistemas Web' tienen funcionalidades muy potentes que brindan respuestas a casos particulares.

Los sistemas Web se pueden utilizar en cualquier navegador Web (Chrome, Firefox, Internet Explorer, etc.) sin importar el sistema operativo. Para utilizar las aplicaciones Web no es necesario instalarlas en cada computadora ya que los usuarios se conectan a un servidor donde se aloja el sistema. (Addappto, 2015, párr. 2)

2.5 Seguimiento

La palabra seguimiento es la acción y efecto de seguir o seguirse, en el contexto popular suele usarse como sinónimo de persecución, observación o vigilancia. Siendo este mismo usado principalmente en el contexto de investigaciones policiales, detectivescas, jurídicas, medicas, científicas, estadística, entre otras; para observar y analizar la evolución un determinado caso. Aunque el término puede aplicarse a cualquier investigación, proceso o proyecto con observación constante. (Concepto Definición, 2021, párr. 1)

2.6 Evaluación por Competencias

Las orientaciones de la Unión Europea insisten en la necesidad de la adquisición de las competencias por parte de la ciudadanía como condición indispensable para lograr que las personas alcancen un pleno desarrollo personal, social y profesional que se ajuste a las demandas de un mundo globalizado y haga posible el desarrollo económico, vinculado al conocimiento.

Las competencias, por tanto, se conceptualizan como un «saber hacer» que se aplica a una diversidad de contextos académicos, sociales y profesionales. Se delimita la definición de competencia, entendida como una combinación de conocimientos, capacidades, o destrezas, y actitudes adecuadas al contexto.

El aprendizaje basado en competencias se caracteriza por su transversalidad, su dinamismo y su carácter integral, el proceso de enseñanza aprendizaje competencial debe abordarse desde todas las áreas de conocimiento y por parte de las diversas instancias que conforman la comunidad educativa, tanto en los ámbitos formales como en los no formales e informales. (Gobierno de Canarias, s.f, párr. 1-3)

Las competencias profesionales son el conjunto integrado de habilidades, conocimientos y aptitudes que se necesitan para desempeñar un empleo específico o desarrollar determinadas actividades profesionales.

Cada puesto de trabajo requiere unas competencias distintas, por lo que, dependiendo de tu objetivo profesional, necesitarás desarrollar unas u otras. Es importante que conozcas las competencias que posees y las que se requerirán en el puesto de trabajo deseado para identificar cuáles son tus áreas de mejora y buscar las mejores opciones para desarrollar esas habilidades. (educaweb, s.f. párr. 1)

2.7 Perfil Profesional

El perfil laboral o profesional es la descripción clara del conjunto de capacidades y competencias que identifican la formación de una persona para encarar responsablemente las funciones y tareas de una determinada profesión o trabajo.

Cuando intentamos conseguir un puesto laboral es importante que podamos transmitir a través de nuestra presentación todo nuestro conocimiento y experiencia para que la persona encargada de la selección de personal se interese por nosotros y nos ofrezca la oportunidad de acceder a la entrevista de trabajo. (argentina.gob.ar, s.f.párr. 1)

El perfil profesional de una persona es la descripción clara del conjunto de capacidades y habilidades que dan muestra de su formación y competencia para afrontar con responsabilidad las funciones y tareas propias de un determinado puesto de trabajo o

de una profesión concreta. O, dicho de otro modo, es una enumeración de las aptitudes laborales de una persona, así como de su formación dentro y fuera del modelo educativo clásico (IOE Business School, s.f. párr.1)

2.8 Metodologías de Desarrollo de software

Las metodologías de desarrollo de software son un conjunto de técnicas y métodos organizativos que se aplican para diseñar soluciones de software informático. El objetivo de las distintas metodologías es el de intentar organizar los equipos de trabajo para que estos desarrollen las funciones de un programa de la mejor manera posible.

Cuando se trata de desarrollar productos o soluciones para un cliente o mercado concreto, es necesario tener en cuenta factores como los costes, la planificación, la dificultad, el equipo de trabajo disponible, los lenguajes utilizados, etc. Todos ellos se engloban en una metodología de desarrollo que permite organizar el trabajo de la forma más ordenada posible.

El desarrollo de software puede ser un sector especialmente complejo, sobre todo cuando se trata de grandes aplicativos y equipos de trabajo. Ponerse a desarrollar un producto sin una metodología clara desembocará en un proceso aún más complejo, que conducirá a problemas, retrasos, errores y, en definitiva, un mal resultado final.

El trabajo con una metodología de desarrollo de software permite reducir el nivel de dificultad, organizar las tareas, agilizar el proceso y mejorar el resultado final de las aplicaciones a desarrollar. (Santander, 2020, párr. 4)

2.8.1 Metodologías agiles

Las metodologías ágiles se basan en el método incremental, en la que en cada ciclo de desarrollo se van agregando nuevas funcionalidades a la aplicación final. Sin embargo, los ciclos son mucho más cortos y rápidos, por lo que se van agregando pequeñas funcionalidades en lugar de grandes cambios.

Este tipo de metodologías permite construir equipos de trabajo autosuficientes e independientes que se reúnen cada poco tiempo para poner en común las novedades.

Poco a poco, se va construyendo y puliendo el producto final, a la vez que el cliente puede ir aportando nuevos requerimientos o correcciones, ya que puede comprobar cómo avanza el proyecto en tiempo real.

Las principales metodologías ágiles son:

- **Kanban:** metodología de trabajo inventada por la empresa de automóviles Toyota. Consiste en dividir las tareas en porciones mínimas y organizarlas en un tablero de trabajo dividido en tareas pendientes, en curso y finalizadas. De esta forma, se crea un flujo de trabajo muy visual basado en tareas prioritarias e incrementando el valor del producto.
- **Scrum:** es también una metodología incremental que divide los requisitos y tareas de forma similar a Kanban. Se itera sobre bloques de tiempos cortos y fijos (entre dos y cuatro semanas) para conseguir un resultado completo en cada iteración. Las etapas son: planificación de la iteración (planning sprint), ejecución (sprint), reunión diaria (daily meeting) y demostración de resultados (sprint review). Cada iteración se denomina sprint.
- **Lean:** está configurado para que pequeños equipos de desarrollo muy capacitados elaboren cualquier tarea en poco tiempo. Los activos más importantes son las personas y su compromiso, relegando así a un segundo plano el tiempo y los costes. El aprendizaje, las reacciones rápidas y potenciar el equipo son fundamentales.
- **Programación extrema (XP):** es una metodología de desarrollo de software basada en las relaciones interpersonales, que se consideran la clave del éxito. Su principal objetivo es crear un buen ambiente de trabajo en equipo y que haya un feedback constante del cliente. El trabajo se basa en 12 conceptos: diseño sencillo, testing, refactorización y codificación con estándares, propiedad colectiva del código, programación en parejas, integración continua, entregas semanales e integridad con el cliente, cliente in situ, entregas frecuentes y planificación. (Santander, 2020)

2.8.2 Metodología XP

La programación extrema o eXtreme Programming (XP) es una metodología de desarrollo de software formulado por Kent Beck, autor del primer libro sobre la materia, Extreme Programming Explained: Embrace Change (1999). Es el más destacado de los procesos ágiles de desarrollo de software. Al igual que éstos XP se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Los defensores de XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos.

Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto y aplicarlo de manera dinámica durante el ciclo de vida del software.

Se fundamenta en modelos de desarrollo de software ágiles porque su filosofía es buscar la satisfacción del cliente y la entrega a corto plazo del software. El equipo XP resalta la entrega sobre el análisis y el diseño y la comunicación activa entre los desarrolladores y los clientes.

Objetivos de XP:

- Satisfacción del cliente.
- Potenciar el trabajo en grupo.
- Minimizar el riesgo actuando sobre las variables del proyecto: costo, tiempo, calidad, alcance.

2.8.2.1 Características de XP

Entre las características principales de la metodología xp tenemos las siguientes:

- Desarrollo iterativo e incremental: Pequeñas mejoras, unas tras otras.

- Realización de pruebas unitarias continuas: Frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se aconseja escribir el código de la prueba antes de la codificación.
- Programación en parejas: Se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto.
- Frecuente integración del equipo de programación con el cliente o usuario: Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.
- Corrección de todos los errores: Antes de añadir una nueva funcionalidad.
- Refactorización del código: Reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad, pero sin modificar su comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.
- Propiedad del código compartida: En vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve que todo el personal pueda corregir y extender cualquier parte del proyecto. Las frecuentes pruebas de regresión garantizan que los posibles errores sean detectados.
- Simplicidad en el código: Es la mejor manera de que las cosas funcionen. Cuando todo funcione, se podrán añadir más funcionalidades si es necesario. La programación extrema apuesta a lo más sencillo, a hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere realizar algo complicado y quizás nunca utilizarlo.

La simplicidad y la comunicación son extraordinariamente complementarias. Con más comunicación resulta más fácil identificar qué se debe y qué no se debe hacer. Cuanto más simple es el sistema, menos tendrá que comunicar sobre éste, lo que lleva a una comunicación más completa, especialmente si se puede reducir el equipo de programadores.

2.8.2.2 Principales roles

Programador: Responsable de implementar las historias de usuario. Además, estima el tiempo de desarrollo de cada historia de usuario para que el cliente pueda asignarle prioridad dentro de la iteración. El programador también es responsable de diseñar y ejecutar los test de unidad del código que ha implementado o modificado y realizar las pruebas unitarias.

Cliente: Escribe las historias de usuario y las pruebas funcionales para validar su implementación. Además, asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración centrándose en aportar mayor valor al negocio. El cliente también es responsable de diseñar y ejecutar los test de aceptación.

Encargado de pruebas (Probador): Ayuda al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.

Encargado de seguimiento (Tracker): Proporciona realimentación al equipo. Verifica el grado de acierto entre las estimaciones realizadas y el tiempo real dedicado, para mejorar futuras estimaciones. Realiza el seguimiento del progreso de cada iteración. Le das seguimiento a la evolución de las estimaciones realizadas por los programadores y compararlas con el tiempo real de desarrollo. De esta forma, puede brindar información estadística en lo que refiere a la calidad de las estimaciones para que puedan ser mejoradas.

Entrenador (Coach): Es responsable del proceso global. Debe proveer de guías al equipo de forma que se apliquen las prácticas XP y se siga el proceso correctamente. Se encarga de iniciar y de guiar a las personas del equipo en poner en marcha cada una de las prácticas de la metodología XP.

Consultor: Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto, en el que puedan surgir problemas.

Gestor (Big boss): Experto en tecnología y labores de gestión. Vínculo entre clientes y programadores. Ayuda a que el equipo trabaje efectivamente creando las

condiciones adecuadas. Su labor esencial es la coordinación. Construye el plantel del equipo, obtiene los recursos necesarios y maneja los problemas que se generan.

2.8.2.3 Proceso XP

El desarrollo es la pieza clave de todo el proceso XP. Todas las tareas tienen como objetivo realizar el desarrollo a la máxima velocidad, sin interrupciones y siempre en la dirección correcta. A grandes rasgos, el ciclo de desarrollo se puede simplificar en los siguientes pasos:

1. El cliente define el valor de negocio a implementar.
2. El programador estima el esfuerzo necesario para su implementación.
3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
4. El programador construye ese valor de negocio.
5. Vuelve al paso 1.

Para Pressman (2010) El procedimiento de XP abarca una serie de normas y métodos que tienen lugar dentro de cuatro etapas o fases fundamentales: planificación, diseño, desarrollo o codificación, y pruebas, las cuales se detallan a continuación.

2.8.2.4 Fases Metodología XP

2.8.2.4.1 Planificación

En la metodología XP, la planificación se concibe como un diálogo constante entre todas las partes implicadas en el proyecto, que abarcan al cliente, los programadores y los coordinadores. El proceso comienza recolectando historias de usuarios, las cuales se equiparán a los convencionales casos de uso. Una vez obtenidas estas historias de usuarios, los programadores llevan a cabo una evaluación ágil del tiempo de desarrollo asociado a cada una. (Meléndez Valladarez, Gaitan, & Pérez Reyes, 2016)

La planificación en XP es iterativa y adaptable, lo que permite responder rápidamente a los cambios en los requisitos del cliente y a los desafíos que puedan surgir durante

el desarrollo. La colaboración continua entre el equipo de desarrollo y el cliente es fundamental para el éxito de la planificación en XP.

Los Conceptos básicos de la planificación son descritos en los siguientes puntos.

Historias de usuario

El sistema es desarrollado para el cliente, por lo tanto el usuario es quien decide que tareas realizará la aplicación. Este planteamiento se desarrolla a lo largo del proyecto: el cliente es quien decide que hacer. Como primer paso, se debe proporcionar una idea clara de lo que será el proyecto en sí. Las historias de usuario son utilizadas como herramienta para dar a conocer los requerimientos al equipo de desarrollo. Son pequeños textos en los que el cliente describe una actividad que realizará el sistema. La redacción de los mismos se realiza bajo las terminologías del cliente, no del desarrollador, de forma que sea clara y sencilla, sin profundizar en detalles.

Se puede considerar que las historias de usuario en XP juegan un papel similar a los casos de uso en otras metodologías, pero en realidad son muy diferentes. Las historias de usuario sólo muestran la silueta de una tarea a realizarse. Por esta razón es fundamental que el usuario o un representante del mismo se encuentren disponibles en todo momento para solucionar dudas, estas no proporcionan información detallada acerca de una actividad específica.

Las historias de usuario tienen las siguientes características:

- **Tarjeta:** En ella se almacena suficiente información para identificar y detallar la historia.
- **Conversación:** Cliente y programadores discuten la historia para ampliar los detalles.
- **Pruebas de Aceptación:** Permite confirmar que la historia ha sido implementada correctamente.

Iteraciones

Las iteraciones en la metodología XP son períodos de tiempo cortos, generalmente de dos a cuatro semanas, durante los cuales el equipo de desarrollo se centra en completar un conjunto de historias de usuario. Las iteraciones se utilizan para dividir

el proyecto en partes más manejables y para obtener comentarios del cliente de forma regular.

Las actividades clave que se realizan durante una iteración de XP son las siguientes:

- **Planificación:** El equipo de desarrollo se reúne para planificar la iteración. Durante la planificación, el equipo identifica las historias de usuario que se completarán durante la iteración y crea un plan para completarlas.
- **Desarrollo:** El equipo de desarrollo desarrolla el software de acuerdo con el plan. Durante el desarrollo, el equipo utiliza las técnicas de XP para crear un software de alta calidad.
- **Pruebas:** El equipo de desarrollo prueba el software para asegurarse de que cumple con los requisitos.
- **Entrega:** El equipo de desarrollo entrega el software al cliente.

Las iteraciones se repiten hasta que el proyecto se completa. A medida que el proyecto avanza, el equipo de desarrollo obtiene comentarios del cliente y utiliza esos comentarios para mejorar el software.

Los beneficios de utilizar iteraciones en XP son los siguientes:

- Se divide el proyecto en partes más manejables. Esto hace que el proyecto sea más fácil de planificar y de gestionar.
- Se obtiene feedback del cliente de forma regular. Esto permite al equipo de desarrollo adaptar el software a las necesidades del cliente.
- Se mejora la calidad del software. El equipo de desarrollo utiliza pruebas para identificar y corregir errores de forma temprana.

Aquí hay algunos consejos para realizar iteraciones en XP:

- Mantenga las iteraciones cortas. Las iteraciones cortas hacen que sea más fácil mantener el enfoque y obtener feedback del cliente.
- Priorice las historias de usuario. El equipo de desarrollo debe priorizar las historias de usuario que son más importantes para el cliente.

- Promueva la comunicación. El equipo de desarrollo debe comunicarse de forma regular para garantizar que todos estén en la misma página.

Las iteraciones son una parte fundamental de la metodología XP. Son una forma eficaz de dividir el proyecto en partes más manejables, obtener feedback del cliente de forma regular y mejorar la calidad del software

Plan de entregas

En la metodología XP, el plan de entregas se basa en la filosofía de ofrecer incrementos de software funcionales de manera constante y sostenible. A continuación, te describo cómo se lleva a cabo un plan de entregas en XP:

También establece que las historias de usuarios serán agrupadas para conformar una entrega y el orden de las mismas. Este cronograma será el resultado de una reunión entre todos los actores del proyecto (Meléndez Valladarez, Gaitan, & Pérez Reyes, 2016, pág. 33)

2.8.2.4.2 Diseño

El diseño de software engloba un conjunto de fundamentos, ideas y métodos que conducen a la creación de un sistema o producto de elevada calidad. Los principios de diseño establecen una filosofía general que orienta la labor de diseño que debe llevarse a cabo. Es esencial comprender los conceptos de diseño antes de implementar sus aspectos mecánicos, y la práctica del diseño en sí conlleva la generación de diversas representaciones del software, sirviendo como referencia para la actividad constructiva que sigue (Pressman, 2010)

El diseño XP sigue rigurosamente el principio MS (mantenlo sencillo). Un diseño siempre se prefiere sobre una presentación más compleja. Además, el diseño guía la implementación de una historia conforme a lo que se escribe. Si el diseño de una historia encuentra un problema de diseño difícil, XP recomienda la creación inmediata de un prototipo de esa porción del diseño. Entonces, se implementa y se evalúa el prototipo de diseño, llamado solución en punta. El objetivo es disminuir el riesgo cuándo comience la implementación verdadera y evaluar las estimaciones originales para la historia que contiene el problema del diseño. En el rediseño el principal objetivo

es controlar dichas modificaciones, sugiriendo pequeños cambios en el diseño que son capaces de mejorarlo en forma radical. Sin embargo, debe notarse que el esfuerzo que requiere el rediseño aumenta en forma notable con el tamaño de la aplicación. (Pressman, 2010, p. 63)

2.8.2.4.3 Codificación

Pressman (2010) indica que después de definir las historias de usuario y realizar un diseño preliminar, el equipo de desarrollo de XP no comienza a codificar inmediatamente. En cambio, desarrolla una serie de pruebas unitarias para cada historia que se incluirá en la próxima entrega.

Las pruebas unitarias ayudan a los desarrolladores a centrarse en lo que necesitan implementar para que el software funcione correctamente. También ayudan a identificar y corregir errores de forma temprana.

Un concepto clave en XP es la programación por parejas. En este enfoque, dos personas trabajan juntas en una estación de trabajo para crear código para una historia de usuario. La programación por parejas tiene varios beneficios, entre ellos:

- Solución de problemas en tiempo real: Dos cabezas piensan mejor que una.
- Aseguramiento de la calidad en tiempo real: El código se revisa conforme se crea.
- Concentración en el problema: Los desarrolladores están menos propensos a distraerse.

En la práctica, cada persona en un equipo de programación por parejas adopta un papel ligeramente diferente. Por ejemplo, una persona puede centrarse en los detalles del código, mientras que la otra se asegura de que el código cumpla con los estándares de codificación.

A medida que los equipos de programación por parejas terminan su trabajo, el código que desarrollan se integra con el trabajo de los demás. Esto se conoce como integración continua. La integración continua ayuda a evitar problemas de compatibilidad e interfaces y a detectar errores temprano.

En resumen, la fase de codificación en XP se centra en la creación de pruebas unitarias, la programación por parejas e la integración continua. Estas técnicas ayudan a los equipos de desarrollo a crear software de alta calidad de forma rápida y eficiente.

Entonces en la fase de codificación se deben tener en cuenta las siguientes características:

- Las pruebas unitarias se crean antes de que se escriba el código. Esto permite a los desarrolladores enfocarse en lo que necesitan implementar para que el software funcione correctamente.
- La programación por parejas se utiliza para mejorar la calidad del código y evitar errores.
- La integración continua se utiliza para detectar errores temprano y evitar problemas de compatibilidad.

2.8.2.4.4 Pruebas

Pressman (2010) Señala que en la metodología XP, las pruebas unitarias son un elemento clave del proceso de desarrollo de software. Las pruebas unitarias se crean antes de que se escriba el código y se automatizan para que puedan ejecutarse repetidamente y con facilidad.

Estas pruebas se utilizan para:

- Identificar y corregir errores de forma temprana.
- Automatizar las pruebas de regresión.
- Proporcionar una indicación continua del progreso del proyecto.

Las pruebas de integración y validación del sistema se realizan a diario, combinando las pruebas unitarias individuales en un "grupo de prueba universal". Esto permite al equipo de desarrollo identificar y corregir errores de forma temprana, antes de que el software se entregue al cliente.

Las pruebas de aceptación son especificadas por el cliente y se centran en las características y funcionalidad generales del sistema. Estas pruebas se derivan de las historias de usuario que se han implementado como parte de la liberación del software.

En resumen, las pruebas son un elemento fundamental de la metodología XP. Las pruebas unitarias, las pruebas de integración y validación del sistema y las pruebas de aceptación ayudan a garantizar que el software sea de alta calidad y cumpla con los requisitos del cliente.

2.8.2.4.5 Lanzamiento

La fase de lanzamiento de la metodología XP es la última fase del proceso de desarrollo. En esta fase, se pone a disposición del usuario el software terminado.

El objetivo del lanzamiento es garantizar que el software cumple con los requisitos del usuario y que está listo para su uso. Para ello, se realizan las siguientes actividades:

- Pruebas de aceptación: Se realizan pruebas con el usuario para garantizar que el software cumple con sus expectativas.
- Documentación: Se genera la documentación del software para ayudar al usuario a utilizarlo.
- Formación: Se proporciona formación al usuario para que pueda utilizar el software.

El lanzamiento de un proyecto de desarrollo de software es un proceso importante que debe realizarse con cuidado. Si el software no cumple con las expectativas del usuario, puede provocar insatisfacción y costes adicionales.

En el contexto de la metodología XP, el lanzamiento se realiza al final de cada iteración. Esto permite al equipo de desarrollo obtener comentarios del usuario de forma regular y ajustar el software en función de esos comentarios.

Las pruebas de aceptación son una parte fundamental del lanzamiento. Estas pruebas se realizan con el usuario real para garantizar que el software cumple con sus necesidades y expectativas. Las pruebas de aceptación pueden ser manuales o automatizadas.

La documentación del software es también importante para el lanzamiento. La documentación debe ser clara y concisa y debe ayudar al usuario a utilizar el software. La documentación puede incluir manuales, guías de usuario, tutoriales y vídeos.

La formación al usuario es otra actividad importante del lanzamiento. La formación debe ser proporcionada por el equipo de desarrollo o por un proveedor externo. La formación debe ayudar al usuario a aprender a utilizar el software de forma eficaz.

En general, el lanzamiento es una fase importante del proceso de desarrollo de software. Si se realiza correctamente, puede ayudar a garantizar que el software cumpla con las expectativas del usuario y que esté listo para su uso.

2.9 Herramientas

2.9.1 Angular

Angular es una plataforma de desarrollo, construida sobre TypeScript como plataforma, Angular incluye:

- Un marco basado en componentes para crear aplicaciones web escalables.
- Una colección de bibliotecas bien integradas que cubren una amplia variedad de funciones, incluido el enrutamiento, la gestión de formularios, la comunicación cliente-servidor y más.
- Un conjunto de herramientas de desarrollo para ayudarlo a desarrollar, compilar, probar y actualizar su código.

Con Angular, está aprovechando una plataforma que puede escalar desde proyectos de un solo desarrollador hasta aplicaciones de nivel empresarial. Angular está diseñado para que la actualización sea lo más sencilla posible, así que aproveche los últimos desarrollos con el mínimo esfuerzo. Lo mejor de todo es que el ecosistema de Angular consta de un grupo diverso de más de 1,7 millones de desarrolladores, autores de bibliotecas y creadores de contenido. (Angular.io, 2022)

Angular es una plataforma que permite desarrollar aplicaciones web en la sección cliente utilizando HTML y JavaScript para que el cliente asuma la mayor parte de la lógica y descargue al servidor con la finalidad de que las aplicaciones ejecutadas a

través de Internet sean más rápidas. El hecho de estar mantenido por Google, así como una serie de innumerables razones técnicas, ha favorecido su rápida adopción por parte de la comunidad de desarrolladores.

Permite la creación de aplicaciones web de una sola página (SPA: single-page application) realizando la carga de datos de forma asíncrona.

Además de mejorar el rendimiento de las aplicaciones web, su utilización en dispositivos móviles está optimizada ya que, en ellos, los ciclos de CPU y memoria son críticos para su óptimo funcionamiento. Ello permite el desarrollo de aplicaciones móviles híbridas con Ionic 2

Gracias al uso de componentes, se puede encapsular mejor la funcionalidad facilitando el mantenimiento de las aplicaciones

Permite el uso de TypeScript (lenguaje desarrollado por Microsoft) con las ventajas que supone poder disponer de un tipado estático y objetos basados en clases. Todo ello, gracias a la especificación ECMAScript 6, que es la base sobre la que se apoya TypeScript. Gracias a un compilador(transpilador) de TypeScript, el código escrito en este lenguaje se traducirá a JavaScript original

En la web Angular de podemos observar el diagrama de arquitectura que muestra cómo se relacionan los elementos principales de una app los cuales son los siguientes:

- Modules.
- Components.
- Templates.
- Metadata.
- Data binding.
- Directives.
- Services.
- Dependency injection (Boada Oriols & Gómez Gutiérrez, 2019)

2.9.2 Nestjs

> Nest (NestJS) is a framework for building efficient, scalable Node.js server-side applications. It uses progressive JavaScript, is built with and fully supports TypeScript (yet still enables developers to code in pure JavaScript) and combines elements of OOP (Object Oriented Programming), FP (Functional Programming), and FRP (Functional Reactive Programming).
>
> Under the hood, Nest makes use of robust HTTP Server frameworks like Express (the default) and optionally can be configured to use Fastify as well!
>
> Nest provides a level of abstraction above these common Node.js frameworks (Express/Fastify), but also exposes their APIs directly to the developer. This gives developers the freedom to use the myriad of third-party modules which are available for the underlying platform. (NestJS , s.f. párr. 1)

Nestjs es un framework de desarrollo basado en Nodejs para aplicaciones del lado del servidor. Nestjs está construido con Typescript y orientado a objetos (POO). Nestjs proporciona una abstracción de los marcos comunes de Nodejs como lo son Express o Fastify.

Nestjs proporciona a los desarrolladores una arquitectura escalable, débilmente acopladas y fácil de mantener. Nestjs en su core utiliza patrones de diseños base para el apoyo de una mejor solución, como lo son el patrón repository e inyección de dependencias. El aporte de Nestjs va desde las diferentes estrategias modulares hasta plugins elaborados por la comunidad, todo esto inspirado en el framework Angular.

2.9.2.1 Características Nestjs

Entre las características principales de framework se tiene:

- Inspirado en Angular: Nestjs promueve todas las buenas prácticas y patrones del framework angular, como lo son la separación de piezas de software a base de módulos, POO, interceptores, pipes, inyección de dependencias, entre otros.

- Ecosistema tecnológico: Nestjs es compatible con las principales tecnologías del desarrollo como GraphQL, bases de datos relacionales, RabbitMQ, Kafka, entre otros. Nestjs proporciona plugins de fácil uso para conectar con diferentes tecnologías.
- Nestjs CLI: Herramienta que nos ayuda con la creación de proyectos, componentes, actualización de dependencias, compilación y generación de código.
- Typescript: Nestjs está construido con TypeScript, agregando tipado, interfaces y otras características para el desarrollo con POO.
- Paquete de pruebas incluido: Nestjs proporciona integración con Jest y Supertest listos para usar en pruebas unitarias, pruebas de integración y e2e.
- Librerías de seguridad: Nestjs proporciona la mayoría de enfoques y estrategias de seguridad, como lo son la autenticación, autorización, encriptación, hashing, CSRF, entre otros. Apoyado y basado en la famosa librería passport. (Esteban, 2022, párr. 1)

2.9.2.2 Conceptos Nestjs

Para la comprensión del framework se debe tener en cuanta los siguiente conceptos

- Controllers: Pieza de software encargada de recepcionar y procesar peticiones HTTP. Los controladores se indentifican con el decorador @Controller() .
- Providers: Pieza de software proveedor de funcionalidades como lo son servicios, fábricas, helpers y otros. La idea principal de los provider es inyectar funcionalidades a toda parte que lo necesite. Los proveedores se identifican con el decorador @Injectable().
- Modules: Pieza de software encargada de agrupar componentes, funcionalidades, encapsular y resolver dependencias. Los módulos se identifican con el decorador @Module().
- Pipes: Pieza de software encargada de realizar transformaciones de datos y validaciones siguiendo el patrón filtro tubería.

- Guards: Pieza de software encargada de determinar si una solicitud es aceptada dependiendo de ciertas condiciones (como permisos, roles, ACL, etc.).
- Interceptor: Pieza de software basado en la tecnica de programación orientada a aspectos (AOP), los interceptores pueden capturar peticiones, agregar lógica adicional antes o después de ejecuciones, ampliar comportamientos de funciones, entre otras.
- Middleware: Pieza de software que provee una capa adicional en el flujo de solicitud y respuesta. El middleware tiene acceso a los objetos request y response de la petición y se pueden apilar uno detrás de otro.
- Inyección de dependencias: Patrón de diseño orientado a objetos que resuelve todas las dependencias que necesita un objeto para ser instanciado. (Esteban, 2022, párr. 10)

2.9.3 Typescript

JavaScript es uno de los lenguajes más populares, en parte porque ha evolucionado y mejorado a pasos agigantados en los últimos años.

Sin embargo, Javascript en algún punto fue un lenguaje que presentaba muchos problemas para bases de código grandes, aplicaciones de gran escala y proyectos con muchos años de desarrollo.

> En 2012 en Javascript no habían clases, ni módulos, el ecosistema carecía de herramientas que optimizaran el flujo de desarrollo, derivado precisamente por las carencias del lenguaje mismo.
>
> 2012 fue el año en que Typescript apareció (luego de 2 años de desarrollo), una solución de Microsoft para el desarrollo de aplicaciones con Javascript a gran escala, para ellos y para sus clientes. Steve Lucco y un equipo de más de 50 personas que incluía a Anders Hejlsberg, Lead Architect de C# y creador de Delphi y Turbo Pascal desarrollaron Typescript en Microsoft, un proyecto que originalmente se conoció como Strada.

Originalmente, productos como Bing y Office 365 despertaron en Microsoft la necesidad de una mejora a JavaScript que permitiera construir productos escalables.

TypeScript es la solución a muchos de los problemas de JavaScript, está pensado para el desarrollo de aplicaciones robustas, implementando características en el lenguaje que nos permitan desarrollar herramientas más avanzadas para el desarrollo de aplicaciones. (Código Facilito, 2018, párr. 1)

2.9.3.1 Características TypeScript

En las características principales tenemos:

- TypeScript es un lenguaje de programación basado en JavaScript que proporciona escritura estática opcional. Esto significa que los IDE pueden proporcionar un entorno más rico para detectar errores de escritura comunes.
- TypeScript puede ser una excelente manera de hacer que el código JavaScript sea más robusto, al tiempo que permite utilizarlo en contextos en los que el JavaScript simple es aceptable.
- TypeScripts es de código abierto y viene con una función inteligente y útil de IntelliSense mientras se escribe; está también disponible para Visual Studio de Microsoft, y para otros entornos de desarrollo (IDE). (ThemeSphere, 2021, párr. 4)

2.9.3.2 Ventajas y Desventajas TypeScript

Ventajas de TypeScript

- TypeScript presenta errores en el momento de la disposición, mientras que JavaScript, en el tiempo de ejecución.
- TypeScript brinda las ventajas de la composición estática discrecional: los tipos TS se pueden agregar a factores, capacidades, propiedades, etc.
- TypeScript sustenta la composición específica o estática. La composición estática puede ser valiosa para ayudar a archivar capacidades, explicar el uso

y reducir la sobrecarga psicológica (pistas de tipo de interfaz y errores esperados al programar continuamente).

- TypeScript se ejecuta en cualquier programa o motor de JavaScript.
- Herramientas extraordinarias con IntelliSense que brinda pistas dinámicas como complemento del código.
- TypeScript ayuda en la organización del código.
- TypeScript tiene una idea de espacio de nombres al caracterizar un módulo.
- Las explicaciones de TypeScript pueden ser discrecionales.
- TypeScript mantiene las interfaces.
- Los módulos de gestión de TypeScript ofrecen una perspectiva destacada entre otros ingenieros de IDE.
- TypeScript tiene una mejor documentación para las API que está en armonía con un código fuente. Algunas organizaciones informan una disminución en los errores cuando cambian a TypeScript.

Desventajas de TypeScript

- TypeScript dedica un gran esfuerzo a incorporar el código.
- TypeScript no admite clases teóricas.
- Cuando se utiliza una biblioteca externa, debe haber un documento de definición, y de vez en cuando generalmente no está disponible.
- La naturaleza de los documentos de definición de tipo es una preocupación.
- Siempre que TypeScript deba ejecutarse en un programa, debería haber un paso de recopilación para cambiar TypeScript a JavaScript.
- TypeScript no es completamente coexpresivo con JavaScript. Los aspectos destacados que faltan incluyen: HOF, Composición, Genéricos con clave más alta. (Barcelona Geeks, 2022, párr. 4-22)

2.9.4 MariaDB

MariaDB es un sistema de gestión de bases de datos que está muy relacionado con MySQL, ya que fue desarrollado por uno de los desarrolladores, Michael “Monty”

Widenius. El objetivo de su desarrollo fue el de mantener el software de gestión de base de datos en un modelo de software libre.

El sistema de gestión de bases de datos MariaDB incorpora las distintas funciones características de MySQL añadiendo algunas mejoras, como la posibilidad de ejecutar consultas complejas y almacenarlas directamente en caché, la nueva gestión de conexiones a BD, la posibilidad de acceder a cluster de datos (interesante para el trabajo en la nube) o el soportar la utilización de jerarquías de graphs y estructuras más complejas.

> En cuanto a seguridad y rendimiento, MariaDB incorpora mejoras, estando siempre en constante evolución gracias a la aportación de una gran comunidad que se encuentra tras de ella. (hostingplus, 2020, párr. 3)

2.9.4.1 Historia MariaDB

Este gestor de base de datos fue desarrollado por uno de los fundadores de la compañía de software MySQL AB: Michael “Monty” Widenius. El proyecto de base de datos MySQL fue adquirido inicialmente por Sun Microsystem hacia febrero del año 2008, manteniéndose como una plataforma open source o de código abierto.

Sin embargo, Michael Widenius temía que la base de datos que había desarrollado pudiera ser comprada y privatizada por otra empresa, por lo que tomó el código fuente original de MySQL y creó MariaDB como su derivado.

Efectivamente Sun Microsystem fue adquirido hacia 2010 por la compañía Oracle Corporation, que hizo que el proyecto MySQL empezara a trabajar bajo una licencia dual de tipo GPL y una Licencia Comercial que le permite poseer el copyright de la mayoría del código e impedir el libre acceso a su código fuente. Widenius, en conjunto con una comunidad de desarrolladores forkearon el proyecto y crearon un nuevo derivado de código abierto llamado MariaDB.

Este servidor, al ser una versión de MySQL, cuenta con todas las funcionalidades de esa base de datos hasta su versión 5.5, además de ciertas características extra como el realizar consultas de un alto nivel de complejidad y poder almacenarlas de manera

directa en el caché, y el uso de jerarquías de graphs y otras estructuras. Además, es posible que se encuentren muchas referencias a MySQL en los ficheros de configuración.

En la actualidad, MariaDB destaca como una de las bases de datos más utilizadas por los usuarios del sistema operativo Linux y Red Hat, así como la mayoría de distribuciones que la incluyen dentro de sus repositorios. (KeepCoding Team, 2022)

2.9.4.2 Características MariaDB

El sistema de gestión de bases de datos MariaDB server se encarga de convertir datos en información estructurada en un gran número de aplicaciones como WordPress, MediaWiki, Drupal, ownCloud, Moodle, entre otras.

Además, cuenta con motores de almacenamiento como Aria, XtraDB, FederateX, SphinxSE, TokuDB, entre otros, siendo los más importantes:

- Aria: es el método de almacenamiento a prueba de fallos, transaccional y no transaccional de MariaDB y funciona como una alternativa al motor MyISAM de MySQL. Usa el caché para almacenar las filas de datos en vez de escribir en disco.
- XtraDB: fue desarrollado para reemplazar al motor de almacenamiento InooDB.

El uso de estos motores, especialmente de Aria, le permite a MariaDB funcionar con una alta velocidad cuando se realizan consultas complejas. Además, este servidor cuenta con una CheckSum Table o Tabla de Suma de Verificación, que tiene un funcionamiento más veloz. MariaDB también redujo el tiempo de conversiones innecesarias de caracteres, lo que, en adición a las características anteriormente mencionadas, han hecho que el sistema pueda funcionar más rápidamente.

Este gestor de bases de datos también añadió extensiones con nuevas funcionalidades como el manejo de hasta 32 segmentos por clave, o el uso del pool de hilos de ejecución o procesos, que permite que MariaDB pueda tener hasta

200.000 conexiones. Otra de las nuevas características es la inclusión del --abort-source-on-error al cliente MySQL. Además, se añade el uso de columnas virtuales, así como un aumento en la precisión de la lista de procesos.

MariaDB también tiene como característica el uso de estadísticas extendidas para el usuario, la selección del motor de almacenamiento y el caché de claves segmentadas. Y destaca por ofrecer mejores medidas de seguridad que otras bases de datos, debido a que incluye elementos como la verificación de contraseña, la autenticación PAM y LDAD, roles de usuario y el cifrado de la base de datos. (KeepCoding Team, 2022, párr. 9)

2.9.5 NodeJs

Node.js es un entorno de tiempo de ejecución de JavaScript (de ahí su terminación en .js haciendo alusión al lenguaje JavaScript). Este entorno de tiempo de ejecución en tiempo real incluye todo lo que se necesita para ejecutar un programa escrito en JavaScript. También aporta muchos beneficios y soluciona muchísimos problemas, por lo que sería más que interesante realizar nuestro curso de Node.js para obtener las bases, conceptos y habilidades necesarias que nos motiven a profundizar en sus opciones e iniciar la programación.

Node.js fue creado por los desarrolladores originales de JavaScript. Lo transformaron de algo que solo podía ejecutarse en el navegador en algo que se podría ejecutar en los ordenadores como si de aplicaciones independientes se tratara. Gracias a Node.js se puede ir un paso más allá en la programación con JavaScript no solo creando sitios web interactivos, sino teniendo la capacidad de hacer cosas que otros lenguajes de secuencia de comandos como Python pueden crear.

Tanto JavaScript como Node.js se ejecutan en el motor de tiempo de ejecución JavaScript V8 (V8 es el nombre del motor de JavaScript que alimenta Google Chrome. Es lo que toma nuestro JavaScript y lo ejecuta mientras navega con Chrome). Este motor coge el código JavaScript y lo convierte en un código de máquina más rápido. El código de máquina es un código de nivel más bajo que la computadora puede

ejecutar sin necesidad de interpretarlo primero, ignorando la compilación y por lo tanto aumentando su velocidad. (Lucas, 2019, párr. 1)

2.9.5.1 Funcionamiento NodeJs

El funcionamiento interno del entorno de ejecución para JavaScript, Node.js, es bastante interesante. En comparación con las técnicas tradicionales de servicio web donde cada conexión (que crea una solicitud) genera un nuevo subproceso, ocupando la RAM del sistema y regularmente maximizando la cantidad de RAM disponible, Node.js opera en un solo subproceso, utilizando el modelo entrada y entrada sin bloqueo de la salida, lo que le permite soportar decenas de miles de conexiones al mismo tiempo mantenidas en el bucle de eventos.

El nodo está completamente controlado por eventos. Resumiendo, podemos decir que el servidor consta de un subproceso que procesa un evento tras otro.

Cuando hay una nueva solicitud se genera un tipo de evento. El servidor empieza a procesarlo y, cuando hay una operación de bloqueo de entrada y salida, no espera hasta que se complete y en su lugar crea una función de devolución de llamada. El servidor comienza en el acto a procesar otro evento (tal vez otra solicitud) y cuando finaliza la operación de entrada y salida, continuará trabajando en la solicitud ejecutando la devolución de llamada tan pronto como tenga tiempo.

Por lo tanto, el servidor nunca necesita crear más subprocesos o cambiar entre subprocesos, lo que significa que tiene muy poca sobrecarga.

Hay que mencionar que en el nivel más bajo de codificación (código C++), encontramos varios subprocesos en Node.js: hay un grupo de eventos de entrada y salida cuyo trabajo es recibir las interrupciones de los propios eventos de entrada y salida y poner los eventos correspondientes en la cola para ser procesados por el hilo principal evitando así la posible interrupción del mismo. (Lucas, 2019, párr. 7)

2.9.5.2 Ventajas NodeJs

En primer lugar y como gran ventaja, es tener Javascript incorporado en la plataforma Node.js, siendo un lenguaje fácil de aprender y que puede ser manejado por programadores de Java.

Node.js se desarrolla en un entorno de tiempo de ejecución de fuente libre que ayudará en el almacenamiento de creación de proyectos únicos.

El modelo de entrada y salida impulsado por eventos ayuda mucho en el manejo simultáneo de peticiones.

El administrador y el usuario incorporan estrategias de codificación similares que desembocan en la creación de abundantes aplicaciones de internet de gran competencia.

Con la implementación de plataformas de desarrollo de software como GitHub Inc., la comunidad Node.js ha crecido de forma exponencial y activa.

Como es un modelo de programación de un solo subproceso, Node.js ofrece abundantes características y opciones. Para tener una buena retención de los eventos de salida en tiempo de ejecución, como las devoluciones de llamada, se implementa Node.js

Node.js es la plataforma de software más utilizada en este momento estando por encima en entornos de ejecución y lenguajes de programación como PHP y C a la vez que tiene un tiempo de ejecución (tiempo real) muy superior.

No es un código muy complejo, pero requiere muchas más líneas de codificación y mayor comprensión que el lenguaje PHP.

El envío de archivos de gran peso también se puede realizar mediante el uso de la tecnología Node.js.

Node.js es una de las tecnologías más usadas hoy en día, y se ha convertido en una de las plataformas más populares utilizadas para el desarrollo de aplicaciones web, aplicaciones de escritorio y servicios. Espero que después de leer este artículo se logre

tener una base de conocimientos básicos que animen a comenzar a escribir sus propias aplicaciones Node.js. (Lucas, 2019, párr. 12)

2.9.6 Git

Git es un sistema de control de versiones distribuido, lo que significa que un clon local del proyecto es un repositorio de control de versiones completo. Estos repositorios locales plenamente funcionales permiten trabajar sin conexión o de forma remota con facilidad. Los desarrolladores confirman su trabajo localmente y, a continuación, sincronizan su copia del repositorio con la copia en el servidor. Este paradigma es distinto del control de versiones centralizado, donde los clientes deben sincronizar el código con un servidor antes de crear nuevas versiones.

La flexibilidad y popularidad de Git hacen que sea una excelente opción para cualquier equipo. Muchos desarrolladores y graduados universitarios ya saben cómo usar Git. La comunidad de usuarios de Git ha creado recursos para entrenar a desarrolladores y la popularidad de Git facilita la ayuda cuando sea necesario. Casi todos los entornos de desarrollo tienen compatibilidad con Git y las herramientas de línea de comandos de Git implementadas en cada sistema operativo principal. (Microsoft, 2023, párr. 2)

2.9.6.1 Conceptos Basicos de Git

Cada vez que se guarda el trabajo, Git crea una confirmación. Una confirmación es una instantánea de todos los archivos en un momento dado. Si un archivo no ha cambiado de una confirmación a la siguiente, Git usa el archivo almacenado anteriormente. Este diseño difiere de otros sistemas que almacenan una versión inicial de un archivo y mantienen un registro de deltas a lo largo del tiempo.

Las confirmaciones crean vínculos a otras confirmaciones, formando un gráfico del historial de desarrollo. Es posible revertir el código a una confirmación anterior, inspeccionar cómo cambiaron los archivos de una confirmación a la siguiente y revisar información como dónde y cuándo se realizaron los cambios. Las confirmaciones se identifican en Git mediante un hash criptográfico único del contenido de la

confirmación. Dado que todo está hash, es imposible realizar cambios, perder información o archivos dañados sin que Git lo detecte.

2.9.6.2 Ramas

Cada desarrollador guarda los cambios en su propio repositorio de código local. Como resultado, puede haber muchos cambios diferentes en función de la misma confirmación. Git proporciona herramientas para aislar los cambios y volver a combinarlos posteriormente. Las ramas, que son punteros ligeros para trabajar en curso, administran esta separación. Una vez finalizado el trabajo creado en una rama, se puede combinar de nuevo en la rama principal (o troncal) del equipo.

2.9.6.3 Ventajas Desarrollo simultáneo

Todos tienen su propia copia local de código y pueden trabajar simultáneamente en sus propias ramas. Git funciona sin conexión, ya que casi todas las operaciones son locales.

Versiones más rápidas: Las ramas permiten el desarrollo flexible y simultáneo. La rama principal contiene código estable y de alta calidad desde el que se publica. Las ramas de características contienen trabajo en curso, que se combinan en la rama principal tras la finalización. Al separar la rama de versión del desarrollo en curso, es más fácil administrar código estable y enviar actualizaciones más rápidamente.

Integración incorporada: Debido a su popularidad, Git se integra en la mayoría de las herramientas y productos. Cada IDE principal tiene compatibilidad integrada con Git y muchas herramientas admiten la integración continua, la implementación continua, las pruebas automatizadas, el seguimiento de elementos de trabajo, las métricas y la integración de características de informes con Git. Esta integración simplifica el flujo de trabajo diario.

Soporte técnico sólido de la comunidad: Git es de código abierto y se ha convertido en el estándar de facto para el control de versiones. No hay escasez de herramientas y recursos disponibles para que los equipos aprovechen. El volumen de compatibilidad

de la comunidad con Git en comparación con otros sistemas de control de versiones facilita la ayuda cuando sea necesario.

Git funciona con cualquier equipo: El uso de Git con una herramienta de administración de código fuente aumenta la productividad de un equipo fomentando la colaboración, aplicando directivas, automatizando procesos y mejorando la visibilidad y la rastreabilidad del trabajo. El equipo puede establecerse en herramientas individuales para el control de versiones, el seguimiento de elementos de trabajo y la integración e implementación continuas. O bien, pueden elegir una solución como GitHub o Azure DevOps que admita todas estas tareas en un solo lugar.

Solicitudes de incorporación de cambios: Use solicitudes de incorporación de cambios para analizar los cambios de código con el equipo antes de combinarlos en la rama principal. Las discusiones en las solicitudes de incorporación de cambios son valiosas para garantizar la calidad del código y aumentar el conocimiento en todo el equipo. Las plataformas como GitHub y Azure DevOps ofrecen una experiencia enriquecida de solicitudes de incorporación de cambios donde los desarrolladores pueden examinar los cambios de archivos, dejar comentarios, inspeccionar confirmaciones, ver compilaciones y votar para aprobar el código.

Directivas de rama: Teams puede configurar GitHub y Azure DevOps para aplicar flujos de trabajo y procesos coherentes en todo el equipo. Pueden configurar directivas de rama para asegurarse de que las solicitudes de incorporación de cambios cumplen los requisitos antes de la finalización. Las directivas de rama protegen ramas importantes mediante la prevención de inserciones directas, la necesidad de revisores y la garantía de compilaciones limpias. de Git (Microsoft, 2023, párr. 4)

2.10 UML

El Lenguaje Unificado de Modelado (UML) fue creado para forjar un lenguaje de modelado visual común y semántica y sintácticamente rico para la arquitectura, el diseño y la implementación de sistemas de software complejos, tanto en estructura como en comportamiento. UML tiene aplicaciones más allá del desarrollo de software, p. ej., en el flujo de procesos en la fabricación.

Es comparable a los planos usados en otros campos y consiste en diferentes tipos de diagramas. En general, los diagramas UML describen los límites, la estructura y el comportamiento del sistema y los objetos que contiene.

UML no es un lenguaje de programación, pero existen herramientas que se pueden usar para generar código en diversos lenguajes usando los diagramas UML. UML guarda una relación directa con el análisis y el diseño orientados a objetos. (Lucid Software Inc, s.f. párr. 1)

2.10.1 Historia UML

"The Three Amigos" (los tres amigos) de la ingeniería de software, como se los conocía, habían desarrollado otras metodologías. Se asociaron para brindar claridad a los programadores creando nuevos estándares. La colaboración entre Grady, Booch y Rumbaugh fortaleció los tres métodos y mejoró el producto final.

Los esfuerzos de estos pensadores derivaron en la publicación de los documentos UML 0.9 y 0.91 en 1996. Pronto se hizo evidente que varias organizaciones, incluidas Microsoft, Oracle e IBM, consideraron que UML era esencial para su propio desarrollo de negocios. Ellos, junto con muchas otras personas y compañías, establecieron los recursos necesarios para desarrollar un lenguaje de modelado hecho y derecho. "Los tres amigos" publicaron la Guía del usuario para el Lenguaje Unificado de Modelado en 1999, y una actualización que incluye información sobre UML 2.0 en la segunda edición de 2005. (Lucid Software Inc, s.f. párr. 7)

2.10.2 La finalidad de UML

Se define los propósitos de UML de la siguiente manera:

- Brindar a arquitectos de sistemas, ingenieros y desarrolladores de software las herramientas para el análisis, el diseño y la implementación de sistemas basados en software, así como para el modelado de procesos de negocios y similares.

- Hacer progresar el estado de la industria permitiendo la interoperabilidad de herramientas de modelado visual de objetos. No obstante, para habilitar un intercambio significativo de información de modelos entre herramientas, se requiere de un acuerdo con respecto a la semántica y notación.
- UML cumple con los siguientes requerimientos:
- Establecer una definición formal de un metamodelo común basado en el estándar MOF (Meta-Object Facility) que especifique la sintaxis abstracta del UML. La sintaxis abstracta define el conjunto de conceptos de modelado UML, sus atributos y sus relaciones, así como las reglas de combinación de estos conceptos para construir modelos UML parciales o completos.
- Brindar una explicación detallada de la semántica de cada concepto de modelado UML. La semántica define, de manera independiente a la tecnología, cómo los conceptos UML se habrán de desarrollar por las computadoras.
- Especificar los elementos de notación de lectura humana para representar los conceptos individuales de modelado UML, así como las reglas para combinarlos en una variedad de diferentes tipos de diagramas que corresponden a diferentes aspectos de los sistemas modelados.
- Definir formas que permitan hacer que las herramientas UML cumplan con esta especificación. Esto se apoya (en una especificación independiente) con una especificación basada en XML de formatos de intercambio de modelos correspondientes (XMI) que deben ser concretados por herramientas compatibles.

2.10.3 Términos de UML

En el desarrollo de UML se usan los siguientes conceptos:

- **Compatibilidad con sintaxis abstracta**: Los usuarios pueden mover modelos a través de diferentes herramientas, incluso si usan diferentes notaciones.
- **Metamodelo de almacén común (CWM):** Interfaces estándares que se usan para permitir el intercambio de metadatos de almacén e inteligencia de negocios

entre herramientas de almacén, plataformas de almacén y repositorios de metadatos de almacén en entornos heterogéneos distribuidos.

- **Compatibilidad con sintaxis concreta**: Los usuarios pueden continuar usando una notación con la que estén familiarizados a través de diferentes herramientas.
- **Núcleo:** En el contexto de UML, el núcleo comúnmente se refiere al "paquete central", que es un metamodelo completo particularmente diseñado para una alta reutilización.
- **Unidad de lenguaje**: Consiste en una colección de conceptos de modelado estrechamente vinculados que proporciona a los usuarios la capacidad de representar aspectos del sistema en estudio según un paradigma o formalismo en particular.
- **Nivel 0 (L0)**: Nivel de cumplimiento inferior para la infraestructura UML - una sola unidad de lenguaje que hace posible el modelado de tipos de estructuras basadas en clases que se encuentran en los lenguajes más populares de programación orientados a objetos.
- **Meta Object Facility (MOF)**: Una especificación de modelado de OMG que brinda la base para las definiciones de metamodelos en la familia de lenguajes MDA de OMG.
- **Metamodelo**: Define el lenguaje y los procesos a partir de los cuales formar un modelo.
- **Construcciones de metamodelos (LM)**: Segundo nivel de cumplimiento en la infraestructura UML - una unidad adicional de lenguaje para estructuras más avanzadas basadas en clases, usadas para construir metamodelos (por medio de CMOF), tales como el UML mismo. UML solo tiene dos niveles de cumplimiento.
- **Arquitectura dirigida por modelos (MDA):** Un enfoque y un plan para lograr un conjunto coherente de especificaciones de tecnología dirigida por modelos.
- **Lenguaje de restricciones para objetos (OCL):** Un lenguaje declarativo para describir reglas que se aplican al Lenguaje Unificado de Modelado. OCL complementa a UML proporcionando términos y símbolos de diagramas de flujo

que son más precisos que el lenguaje natural, pero menos difíciles de dominar que las matemáticas.

- **Object Management Group (OMG):** Es un consorcio sin fines de lucro de especificaciones para la industria de la computación, cuyos miembros definen y mantienen la especificación UML.
- **UML 1**: Primera versión del Lenguaje Unificado de Modelado.
- **Lenguaje Unificado de Modelado (UML)**: Un lenguaje visual para especificar, construir y documentar los artefactos de los sistemas.
- **XMI**: Una especificación basada en XML de formatos de intercambio de modelos correspondientes.

2.10.4 Tipos de diagramas UML

UML usa elementos y los asocia de diferentes formas para formar diagramas que representan aspectos estáticos o estructurales de un sistema, y diagramas de comportamiento, que captan los aspectos dinámicos de un sistema.

Diagramas UML estructurales

- **Diagrama de clases:** El diagrama UML más comúnmente usado, y la base principal de toda solución orientada a objetos. Las clases dentro de un sistema, atributos y operaciones, y la relación entre cada clase. Las clases se agrupan para crear diagramas de clases al crear diagramas de sistemas grandes.
- **Diagrama de componentes:** Muestra la relación estructural de los elementos del sistema de software, muy frecuentemente empleados al trabajar con sistemas complejos con componentes múltiples. Los componentes se comunican por medio de interfaces.
- **Diagrama de estructura compuesta:** Los diagramas de estructura compuesta se usan para mostrar la estructura interna de una clase.
- **Diagrama de implementación:** Ilustra el hardware del sistema y su software. Útil cuando se implementa una solución de software en múltiples máquinas con configuraciones únicas.

- **Diagrama de objetos:** Muestra la relación entre objetos por medio de ejemplos del mundo real e ilustra cómo se verá un sistema en un momento dado. Dado que los datos están disponibles dentro de los objetos, estos pueden usarse para clarificar relaciones entre objetos.
- **Diagrama de paquetes:** Hay dos tipos especiales de dependencias que se definen entre paquetes: la importación de paquetes y la fusión de paquetes. Los paquetes pueden representar los diferentes niveles de un sistema para revelar la arquitectura. Se pueden marcar las dependencias de paquetes para mostrar el mecanismo de comunicación entre niveles.
- Diagramas UML de comportamiento
- **Diagramas de actividades:** Flujos de trabajo de negocios u operativos representados gráficamente para mostrar la actividad de alguna parte o componente del sistema. Los diagramas de actividades se usan como una alternativa a los diagramas de máquina de estados.
- **Diagrama de comunicación**: Similar a los diagramas de secuencia, pero el enfoque está en los mensajes que se pasan entre objetos. La misma información se puede representar usando un diagrama de secuencia y objetos diferentes.
- **Diagrama de panorama de interacciones**: Hay siete tipos de diagramas de interacciones. Este diagrama muestra la secuencia en la cual actúan.
- **Diagrama de secuencia:** Muestra cómo los objetos interactúan entre sí y el orden de la ocurrencia. Representan interacciones para un escenario concreto.
- **Diagrama de máquina de estados:** Similar a los diagramas de actividades, describen el comportamiento de objetos que se comportan de diversas formas en su estado actual.
- **Diagrama de temporización:** Al igual que en los diagramas de secuencia, se representa el comportamiento de los objetos en un período de tiempo dado. Si hay un solo objeto, el diagrama es simple. Si hay más de un objeto, las interacciones de los objetos se muestran durante ese período de tiempo particular.

- **Diagrama de caso de uso:** Representa una funcionalidad particular de un sistema. Se crea para ilustrar cómo se relacionan las funcionalidades con sus controladores (actores) internos/externos (Lucid Software Inc, s.f. párr. 53)

2.11 Calidad de Software

La calidad del software es el conjunto de cualidades que lo caracterizan y que determinan su utilidad y existencia. La calidad es sinónimo de eficiencia, flexibilidad, corrección, confiabilidad, mantenibilidad, portabilidad, usabilidad, seguridad e integridad.

La calidad del software es medible y varía de un sistema a otro o de un programa a otro. Un software elaborado para el control de naves espaciales debe ser confiable al nivel de "cero fallas"; un software hecho para ejecutarse una sola vez no requiere el mismo nivel de calidad; mientras que un producto de software para ser explotado durante un largo período (10 años o más), necesita ser confiable, mantenible y flexible para disminuir los costos de mantenimiento y perfeccionamiento durante el tiempo de explotación.

La calidad del software puede medirse después de elaborado el producto. Pero esto puede resultar muy costoso si se detectan problemas deriva dos de imperfecciones en el diseño, por lo que es imprescindible tener en cuenta tanto la obtención de la calidad como su control durante todas las etapas del ciclo de vida del software. (Fernández Carrasco, García León, & Beltrán Benavides, 1995, párr. 7)

2.11.1 Norma ISO/IEC 25000

ISO 25000, conocida como SQuaRE (System and Software Quality Requirements and Evaluation), es una familia de normas que tiene por objetivo la creación de un marco de trabajo común para evaluar la calidad del producto software.

Actualmente es una familia de normas ISO (aprobada por la Organización Internacional de Normalización) y el IEC (Comisión Internacional Electrotécnica).

Obtener esta certificación de calidad proporciona a las empresas que desarrollan software el reconocimiento externo de la calidad de los productos de software que desarrollan.

Además, también beneficia a las organizaciones que se dedican a la compra de software ya que pueden evaluar dicha calidad y conocer si se adaptaría el producto a sus necesidades. (ACMS, 2022)

2.11.1.1 ISO/IEC 25010

El modelo de calidad representa la piedra angular en torno a la cual se establece el sistema para la evaluación de la calidad del producto. En este modelo se determinan las características de calidad que se van a tener en cuenta a la hora de evaluar las propiedades de un producto software determinado.

La calidad del producto software se puede interpretar como el grado en que dicho producto satisface los requisitos de sus usuarios aportando de esta manera un valor. Son precisamente estos requisitos (funcionalidad, rendimiento, seguridad, mantenibilidad, etc.) los que se encuentran representados en el modelo de calidad, el cual categoriza la calidad del producto en características y subcaracterísticas. (iso25000, 2022)

2.11.1.2 Adecuación Funcional

Representa la capacidad del producto software para proporcionar funciones que satisfacen las necesidades declaradas e implícitas, cuando el producto se usa en las condiciones especificadas. Esta característica se subdivide a su vez en las siguientes subcaracterísticas:

- Completitud funcional. Grado en el cual el conjunto de funcionalidades cubre todas las tareas y los objetivos del usuario especificados.
- Corrección funcional. Capacidad del producto o sistema para proveer resultados correctos con el nivel de precisión requerido.

- Pertinencia funcional. Capacidad del producto software para proporcionar un conjunto apropiado de funciones para tareas y objetivos de usuario especificados.

2.11.1.3 Eficiencia de desempeño

Esta característica representa el desempeño relativo a la cantidad de recursos utilizados bajo determinadas condiciones. Esta característica se subdivide a su vez en las siguientes subcaracterísticas:

- Comportamiento temporal. Los tiempos de respuesta y procesamiento y los ratios de throughput de un sistema cuando lleva a cabo sus funciones bajo condiciones determinadas en relación con un banco de pruebas (benchmark) establecido.
- Utilización de recursos. Las cantidades y tipos de recursos utilizados cuando el software lleva a cabo su función bajo condiciones determinadas.
- Capacidad. Grado en que los límites máximos de un parámetro de un producto o sistema software cumplen con los requisitos.

2.11.1.4 Usabilidad

Capacidad del producto software para ser entendido, aprendido, usado y resultar atractivo para el usuario, cuando se usa bajo determinadas condiciones. Esta característica se subdivide a su vez en las siguientes subcaracterísticas:

- **Reconocibilidad de la adecuación.** Capacidad del producto que permite al usuario entender si el software es adecuado para sus necesidades.
- **Aprendizabilidad.** Capacidad del producto que permite al usuario aprender su aplicación.
- **Operabilidad.** Capacidad del producto que permite al usuario operarlo y controlarlo con facilidad.
- **Protección contra errores de usuario.** Capacidad del sistema para proteger a los usuarios de hacer errores.

- **Estética de la interfaz de usuario.** Capacidad de la interfaz de usuario de agradar y satisfacer la interacción con el usuario.
- **Accesibilidad.** Capacidad del producto que permite que sea utilizado por usuarios con determinadas características y discapacidades.

2.11.1.5 Fiabilidad

Capacidad de un sistema o componente para desempeñar las funciones especificadas, cuando se usa bajo unas condiciones y periodo de tiempo determinados. Esta característica se subdivide a su vez en las siguientes subcaracterísticas:

- **Madurez.** Capacidad del sistema para satisfacer las necesidades de fiabilidad en condiciones normales.
- **Disponibilidad.** Capacidad del sistema o componente de estar operativo y accesible para su uso cuando se requiere.
- **Tolerancia a fallos.** Capacidad del sistema o componente para operar según lo previsto en presencia de fallos hardware o software.
- **Capacidad de recuperación.** Capacidad del producto software para recuperar los datos directamente afectados y reestablecer el estado deseado del sistema en caso de interrupción o fallo

2.11.1.6 Seguridad

Capacidad de protección de la información y los datos de manera que personas o sistemas no autorizados no puedan leerlos o modificarlos. Esta característica se subdivide a su vez en las siguientes subcaracterísticas:

- **Confidencialidad.** Capacidad de protección contra el acceso de datos e información no autorizados, ya sea accidental o deliberadamente.
- **Integridad.** Capacidad del sistema o componente para prevenir accesos o modificaciones no autorizados a datos o programas de ordenador.

- **No repudio.** Capacidad de demostrar las acciones o eventos que han tenido lugar, de manera que dichas acciones o eventos no puedan ser repudiados posteriormente.
- **Responsabilidad.** Capacidad de rastrear de forma inequívoca las acciones de una entidad.
- **Autenticidad.** Capacidad de demostrar la identidad de un sujeto o un recurso.

2.11.1.7 Mantenibilidad

Esta característica representa la capacidad del producto software para ser modificado efectiva y eficientemente, debido a necesidades evolutivas, correctivas o perfectivas. Esta característica se subdivide a su vez en las siguientes subcaracterísticas:

- **Modularidad.** Capacidad de un sistema o programa de ordenador (compuesto de componentes discretos) que permite que un cambio en un componente tenga un impacto mínimo en los demás.
- **Reusabilidad.** Capacidad de un activo que permite que sea utilizado en más de un sistema software o en la construcción de otros activos.
- **Analizabilidad.** Facilidad con la que se puede evaluar el impacto de un determinado cambio sobre el resto del software, diagnosticar las deficiencias o causas de fallos en el software, o identificar las partes a modificar.
- **Capacidad para ser modificado.** Capacidad del producto que permite que sea modificado de forma efectiva y eficiente sin introducir defectos o degradar el desempeño.
- **Capacidad para ser probado.** Facilidad con la que se pueden establecer criterios de prueba para un sistema o componente y con la que se pueden llevar a cabo las pruebas para determinar si se cumplen dichos criterios.

2.11.1.8 Portabilidad

Capacidad del producto o componente de ser transferido de forma efectiva y eficiente de un entorno hardware, software, operacional o de utilización a otro. Esta característica se subdivide a su vez en las siguientes sus características:

- **Adaptabilidad.** Capacidad del producto que le permite ser adaptado de forma efectiva y eficiente a diferentes entornos determinados de hardware, software, operacionales o de uso.
- **Capacidad para ser instalado.** Facilidad con la que el producto se puede instalar y/o desinstalar de forma exitosa en un determinado entorno.
- **Capacidad para ser reemplazado.** Capacidad del producto para ser utilizado en lugar de otro producto software determinado con el mismo propósito y en el mismo entorno. (iso25000, 2022)

2.12 Estimación de costo del desarrollo de Software

Una estimación de software es una predicción de cuánto tiempo durará o costará su desarrollo y mantenimiento. Si se trata de una estimación de tiempo, el esfuerzo puede expresarse en horas-persona u otra unidad, si se trata de estimación de costo, se puede expresar en la moneda de preferencia.

El reto de elaborar estimaciones de software, es realizar predicciones realistas, basándose en información incompleta e incierta.

En Ingeniería de software y gestión de proyectos de software, las estimaciones se utilizan para:

- Desarrollar planes de proyectos.
- Elaborar planificaciones de iteración en desarrollo de software.
- Elaborar presupuestos.
- Realizar análisis de inversión.
- Fijación de precios de un software para un cliente empresarial.
- Análisis para determinar el precio en software dirigido al consumidor.

Para planificar estrategia cuando se dispone a participar en subastas de contratos en los que participan varios proveedores. (PMOinformatica.com, 2018, párr. 4)

2.12.1 COCOMO II

Las siglas COCOMO significan Constructive Cost Model, o “Modelo constructivo de costos” en español (COCOMO II). El método COCOMO fue desarrollado originalmente por el Dr. Barry Boehm en 1981. Luego en los años 90 se realizó una actualización y a partir de 1995 se conoce como COCOMO II.

COCOMO II está documentado en el Manual COCOMO II, disponible en el sitio web de la Universidad del Sur de California (University of Southern California).

Para realizar estimaciones de proyectos de software, COCOMO II utiliza tres submodelos, cada uno con mayor nivel de fidelidad que el anterior, estos modelos son:

- Composición de aplicaciones.
- Diseño inicial (Diseño temprano).
- Modelos post arquitectura.

El Modelo de composición de aplicaciones se utiliza para analizar el software a desarrollar y modificar, identificando componentes, subcomponentes, dividiéndolos y clasificándolos. Los modelos de diseño inicial y post arquitectura son los que producen la estimación, ambos utilizan las siguientes ecuaciones:

Esfuerzo expresado en personas meses (E)= a (Kl)^b x m (X)

donde:

a, b: Constantes con valores asignados según el modelo que se esté utilizando (Diseño inicial o post arquitectura).

Kl: Líneas de código (em miles).

m (X): Multiplicador que depende de 15 atributos.

Tiempo (Calendario) del proyecto (Tdev) = c (E)^d

donde:

c, d: Constantes con valores asignados según el modelo que se esté utilizando (Diseño inicial o post arquitectura).

- E: Esfuerzo requerido (Calculado con la primera ecuación)
- Número de personas requeridas en el equipo (P) = E / Tdev

La variable clave para lograr una estimación acertada en el modelo, es la de medición del tamaño del software a desarrollar. Para la medición del tamaño, COCOMO propone dos enfoques, el conteo de líneas de código y el conteo de puntos de función.

2.12.1.1 COCOMO II con conteo de líneas de código

El mayor reto es poder calcular cuantas líneas de código tendrá un proyecto que aún no hemos realizado, COCOMO II propone el uso de datos históricos que se tengan de proyectos pasados.

Si no se tienen datos históricos, se puede recurrir a la opinión de expertos solo para obtener una medición de líneas de código, siendo recomendable tener 3 estimaciones, una pesimista, más probable y optimista.

Como se puede ver, COCOMO II también se vale del juicio de expertos, sin embargo, tener una fórmula matemática probada para establecer una correlación entre las líneas de código probables y el esfuerzo requerido es mejor que simplemente valernos de la opinión del estimador. (PMOinformatica.com, 2018, párr. 78)

2.12.1.2 COCOMO II con puntos de función

La estimación de costos por puntos de función se basa en la cantidad de funcionalidad involucrada en el software a desarrollar, son útiles porque se basan en información que está disponible desde que se realiza el análisis del proyecto.

Los puntos de función no ajustados miden un proyecto de software por medio de la cuantificación de las funcionalidades de procesamiento de información, clasificando las mismas en Entradas o salidas de datos, archivos lógicos internos o externos. Para cada uno se asigna una cantidad predeterminada de puntos de función, que nos aporta una medición funcional del software.

COCOMO II sigue las mismas definiciones el método de puntos de función IFPUG, el cual veremos en más detalle en la siguiente técnica de estimación de proyectos de software. (PMOinformatica.com, 2018, párr. 90)

2.13 Seguridad

2.13.1 Seguridad del Software

La seguridad del software es el concepto de implementar mecanismos en la construcción de la seguridad para ayudarla a permanecer funcional (o resistente) a los ataques. Esto significa que una pieza de software se somete a pruebas de seguridad antes de salir al mercado para comprobar su capacidad para resistir ataques maliciosos.

La idea detrás de la seguridad del software es crear software que sea seguro desde el principio sin tener que agregar elementos de seguridad adicionales para agregar capas adicionales de seguridad (aunque en muchos casos, esto todavía sucede). El siguiente paso es enseñar a los usuarios a usar el software de la manera correcta para evitar ser propensos o estar expuestos a ataques.

La seguridad del software es fundamental porque un ataque de malware puede causar un daño extremo a cualquier pieza de software al tiempo que compromete la integridad, la autenticación y la disponibilidad. Si los programadores tienen esto en cuenta en la etapa de programación y no después, el daño puede detenerse antes de que comience. (Thales, s.f. párr. 4)

2.13.2 Seguridad de la Información

Estamos ante un proceso en que se está produciendo modificaciones, por lo que el término Seguridad de la Información está tomando una traducción más acertada sobre information security. Aunque aún hay muchos especialistas que siguen nombrándolo según el enfoque técnico que hemos comentado anteriormente.

La Seguridad de la Información es muy extensa, por lo que no es sólo una cuestión técnica, sino que supone una responsabilidad de la alta dirección de la empresa, así como de sus directivos.

Tenemos que tomar en cuenta que el ambiente TIC está orientado al servicio y a la actuación en función de los procesos de negocio. Se diferencia de los procesos centrales de la misma empresa que constituyen el núcleo de los negocios de la empresa.

En el caso de no involucrarse las unidades activas y los líderes de negocio, como podrían ser, ejecutivos, directivos, etc. de las entidades, no podrá existir un plan de Seguridad de la Información, a partir de todos los riesgos determinados. Todo ello se lleva a cabo en el seno del sistema de dirección y control propio del gobierno corporativo.

Se tiene que considerar los sujetos, los procesos y las funciones de negocio, además de la protección de todos los activos/recursos de la entidad impulsora, propietaria y beneficiaria de la Seguridad de la Información, dentro de un marco de responsabilidades compartidas.

Se tienen que considerar la totalidad de los riesgos técnicos de TIC, además de que la seguridad se desarrolle por toda la empresa, es decir, son riesgos organizacionales, operacionales y físicos.

Los riesgos operacionales son hoy en día más cruciales en lo referente a Seguridad de la Información. Las vulnerabilidades de este tipo de riesgo se expanden durante una amplia gama de grises, en conexión con el comportamiento humano y los juicios subjetivos de las personas, la resistencia al cambio, la cultura empresarial, la forma de comunicarse, etc. (isotools, s.f. párr. 11)

2.13.3 ISO 27001:2013

El eje central de ISO 27001:2013 es proteger la confidencialidad, integridad y disponibilidad de la información en una empresa. Esto lo hace investigando cuáles son los potenciales problemas que podrían afectar la información (es decir, la evaluación

de riesgos) y luego definiendo lo que es necesario hacer para evitar que estos problemas se produzcan (es decir, mitigación o tratamiento del riesgo).

Por lo tanto, la filosofía principal de la norma ISO 27001:2013 se basa en la gestión de riesgos: investigar dónde están los riesgos y luego tratarlos sistemáticamente.

> Las medidas de seguridad (o controles) que se van a implementar se presentan, por lo general, bajo la forma de políticas, procedimientos e implementación técnica (por ejemplo, software y equipos). Sin embargo, en la mayoría de los casos, las empresas ya tienen todo el hardware y software, pero utilizan de una forma no segura; por lo tanto, la mayor parte de la implementación de ISO 27001:2013 estará relacionada con determinar las reglas organizacionales (por ejemplo, redacción de documentos) necesarias para prevenir violaciones de la seguridad.
>
> Como este tipo de implementación demandará la gestión de múltiples políticas, procedimientos, personas, bienes, etc., ISO 27001:2013 ha detallado cómo amalgamar todos estos elementos dentro del sistema de gestión de seguridad de la información (SGSI).
>
> Por eso, la gestión de la seguridad de la información no se acota solamente a la seguridad de TI (por ejemplo, cortafuegos, anti-virus, etc.), sino que también tiene que ver con la gestión de procesos, de los recursos humanos, con la protección jurídica, la protección física, etc. (Advisera, s.f. párr. 4)

2.13.4 Sistema de Gestión de Seguridad de la Información

El Sistema de Gestión de Seguridad de la Información (SGSI) es el concepto central sobre el que se construye la norma ISO 27001:2013 De acuerdo con esta normativa, la seguridad de la información consiste en la preservación de la confidencialidad, integridad y disponibilidad y es bajo estos tres términos que se realiza el análisis y evaluación de los activos de información.

El SGSI debe estar enfocado en tres fundamentos:

2.13.5 Disponibilidad

La disponibilidad se garantiza mejor manteniendo rigurosamente todo el hardware, realizando reparaciones de hardware de inmediato cuando sea necesario y manteniendo un entorno de sistema operativo que funcione correctamente y libre de conflictos de software. También es importante mantenerse al día con todas las actualizaciones necesarias del sistema.

Proporcionar un ancho de banda de comunicación adecuado y prevenir la aparición de cuellos de botella son igualmente importantes. La redundancia, la conmutación por error, RAID, incluso los clústeres de alta disponibilidad pueden mitigar las graves consecuencias cuando se producen problemas de hardware.

La recuperación ante desastres rápida y adaptativa es esencial para los peores escenarios; esa capacidad depende de la existencia de un plan integral de recuperación ante desastres (DRP). Las salvaguardas contra la pérdida de datos o las interrupciones en las conexiones deben incluir eventos impredecibles como desastres naturales e incendios. Para evitar la pérdida de datos de tales ocurrencias, se puede almacenar una copia de seguridad en una ubicación aislada geográficamente, tal vez incluso en una caja fuerte a prueba de fuego e impermeable. Los equipos o software de seguridad adicionales, como los servidores de seguridad y los servidores proxy, pueden proteger contra el tiempo de inactividad y los datos inaccesibles debido a acciones maliciosas, como ataques de denegación de servicio (DoS) e intrusiones en la red.

2.13.6 Confidencialidad

La confidencialidad es aproximadamente equivalente a la privacidad. Las medidas emprendidas para garantizar la confidencialidad están diseñadas para evitar que la información confidencial llegue a las personas equivocadas, al tiempo que se garantiza que las personas adecuadas puedan obtenerla: el acceso debe estar restringido a aquellos autorizados para ver los datos en cuestión. También es común que los datos se clasifiquen de acuerdo con la cantidad y el tipo de daño que se podría hacer si cae

en manos no deseadas. Se pueden implementar medidas más o menos estrictas de acuerdo con esas categorías.

2.13.7 Integridad

La integridad implica mantener la consistencia, precisión y confiabilidad de los datos durante todo su ciclo de vida. Los datos no deben modificarse en tránsito, y deben tomarse medidas para garantizar que personas no autorizadas no puedan alterar los datos (por ejemplo, en una violación de la confidencialidad). Estas medidas incluyen permisos de archivos y controles de acceso de usuarios.

El control de versiones puede usarse para evitar que los cambios erróneos o la eliminación accidental por parte de usuarios autorizados se conviertan en un problema. Además, deben existir algunos medios para detectar cualquier cambio en los datos que pueda ocurrir como resultado de eventos no causados por el hombre, como un pulso electromagnético (EMP) o un bloqueo del servidor. Algunos datos pueden incluir sumas de verificación, incluso sumas de verificación criptográficas, para verificar la integridad. Las copias de seguridad o redundancias deben estar disponibles para restaurar los datos afectados a su estado correcto. (pirani, s.f. párr. 57)

2.14 Pruebas de Software

La prueba de software es el proceso de evaluar y verificar que un producto o aplicación de software hace lo que se supone que debe hacer. Los beneficios de las pruebas incluyen la prevención de errores, la reducción de los costos de desarrollo y la mejora del rendimiento. (IBM, s.f. párr. 1)

Dentro de las pruebas de software se tienen los siguientes:

2.14.1 Prueba unitaria

La prueba unitaria es un método de prueba que se centra en examinar “unidades” individuales o fragmentos de código.

El objetivo principal de las pruebas unitarias es determinar la integridad lógica: que un fragmento de código haga lo que se supone que debe hacer.

Generalmente, la gente probará métodos o funciones individuales como unidades y, dependiendo del tamaño y complejidad del código, también clases. Se prueban de forma aislada y, posteriormente, las dependencias típicas se eliminan o se burlan.

Un ejemplo de esto sería si tuviera una función que masajee los datos de una base de datos. Sin embargo, dado que es una prueba unitaria, no usaría una base de datos real: haría una llamada a un punto final con stub, que devuelve los datos que normalmente esperaría de una base de datos. De esa forma, la única funcionalidad que se está probando es este fragmento de código o la unidad.

2.14.2 Prueba de integración

Las pruebas de integración son un método de prueba centrado en examinar varios componentes juntos.

El objetivo principal de las pruebas de integración es garantizar la integridad de la relación y el flujo de datos entre componentes o unidades.

Por lo general, las personas ejecutarán primero pruebas unitarias para probar la integridad lógica de las unidades individuales. Luego, ejecutarán pruebas de integración para garantizar que la interacción entre estas unidades se comporte como se esperaba. Continuando con el ejemplo anterior, una prueba de integración en este caso sería ejecutar la misma prueba contra una base de datos real. Con bases de datos reales, tiene escenarios y comportamientos adicionales a considerar.

“Prueba de integración” es un término amplio y abarca cualquier prueba en la que estén involucrados múltiples componentes. Posteriormente, se puede usar una gran variedad de tecnologías y marcos, incluidos los mismos que se usaron anteriormente en pruebas unitarias, o marcos separados basados en el comportamiento (los ejemplos se enumeran en la siguiente sección).

2.14.3 Prueba de aceptación

Las pruebas de aceptación suelen ser una fase del ciclo de desarrollo, el objetivo principal de las pruebas de aceptación es verificar que un producto o característica determinada se haya desarrollado de acuerdo con las especificaciones establecidas por un cliente o un interesado interno, como un gerente de producto.

Dentro de las pruebas de aceptación, también puede haber múltiples fases, como las pruebas α o las pruebas β. A medida que gran parte del mundo del desarrollo de software avanza hacia los procesos ágiles, las pruebas de aceptación del usuario se han vuelto mucho menos rígidas y más colaborativas.

Es importante tener en cuenta que, si bien las pruebas de aceptación pueden verificar que la aplicación se comporte como el usuario desea, no verifica la integridad del sistema. Otra advertencia de las pruebas de aceptación del usuario es que hay un límite para los casos y escenarios de la esquina que una persona puede idear; es por eso que los métodos de prueba automatizados anteriores son importantes ya que cada caso de uso y escenario está codificado.

2.14.4 Prueba de caja blanca (estructural, caja transparente)

Las pruebas de caja blanca (también llamadas estructurales o de caja transparente) describen pruebas o métodos en los que se conocen los detalles y el funcionamiento interno del software que se está probando.

Dado que conoce las funciones, los métodos, las clases, cómo funcionan y cómo se unen, generalmente está mejor equipado para examinar la integridad lógica del código.

Por ejemplo, es posible que sepa que hay una peculiaridad en la forma en que un determinado idioma maneja ciertas operaciones. Podría escribir pruebas específicas para eso, que de otro modo no sabría escribir en un escenario de caja negra.

Las pruebas unitarias y las pruebas de integración suelen ser una caja blanca.

2.14.5 Pruebas de caja negra (funcional, conductual, caja cerrada)

Por el contrario, las pruebas de caja negra (también llamadas funcionales, de comportamiento o de caja cerrada) describen cualquier prueba o método en el que se desconocen los detalles y el funcionamiento interno del software que se está probando.

Dado que no conoce ninguno de los detalles, realmente no puede crear casos de prueba que se dirijan a escenarios de nicho específicos o que hagan hincapié en la lógica específica del sistema.

Lo único que sabe es que, para una solicitud o una entrada determinada, se espera un determinado comportamiento o salida. Por lo tanto, las pruebas de caja negra prueban principalmente el comportamiento de un sistema. Las pruebas de un extremo a otro suelen ser una caja negra.

2.14.6 Pruebas visuales / de interfaz de usuario (pruebas de navegador)

Las pruebas de la interfaz de usuario o del navegador describen pruebas que examinan específicamente la integridad y el comportamiento de los componentes de la interfaz de usuario.

A menudo, cuando se utiliza un sitio web, se espera que determinadas acciones den lugar a determinados estados. Las pruebas de IU verifican que esto ocurra correctamente. Por ejemplo, la forma en que ha implementado cierto CSS puede fallar en Firefox, pero no en Chrome. Las pruebas del navegador pueden comprobarlo.

Hay muchos marcos de prueba de navegadores populares, como Selenium , Cypress , TestCafe , SauceLabs , Katalon Studio , Browsersync , Robot , etc.

2.14.7 Prueba de estrés

Las pruebas de carga se refieren a probar la respuesta de una aplicación al aumento de la demanda.

Esto incluye probar las afluencias repentinas de solicitudes o usuarios que podrían ejercer una presión inesperada en el sistema. Las pruebas de carga a menudo se

realizan como parte de las pruebas de seguridad para garantizar que una aplicación y su sistema no puedan ser DDOS.

Las pruebas de carga también se realizan para verificar la cantidad máxima de datos que un sistema puede manejar en un momento dado. Es fundamental para ayudar a los equipos a determinar fórmulas de escalamiento e implementación de alta disponibilidad (HA) efectivas.

2.14.8 Pruebas de inserción

Las pruebas de inserción son una forma de prueba de seguridad que implica verificar la solidez de la seguridad de una aplicación.

Se explotan todas las formas en que una aplicación puede verse comprometida (secuencias de comandos entre sitios, entradas no desinfectadas, ataques de desbordamiento de búfer, etc.) para comprobar cómo la maneja el sistema. Las pruebas de penetración son una parte importante para asegurarse de que una empresa no sea víctima de infracciones graves. (DEVOPS LATAM, 2021, párr. 2)

MARCO APLICATIVO

3.1 Introducción

En el presente capitulo tiene como finalidad describir el análisis, diseño, gestión, evaluación de métricas de calidad y costo estimado del proyecto, se aplicará los conceptos mencionados en el capítulo II haciendo uso de las herramientas para el correcto desarrollo del análisis y diseño del sistema aplicando las fases descritas en la metodología XP (Extreme Programing), así de esta manera siguiendo los parámetros descritos en la metodología xp se lograra producir un software de calidad todo lo anterior apoyado con el modelado UML.

3.1 Fase de planificación

Primera fase la metodología XP (Extreme Programing). donde se obtuvieron datos para hacer el relevamiento de requerimientos a través de una serie de entrevistas con la comisión encargada de la carrera de Medicina de la universidad Pública de el Alto para el seguimiento de sistema e implementación.

3.1.1 Requerimiento Funcionales

Los requerimientos obtenidos en las entrevistas son:

- Los estudiantes pueden ver datos acerca de su progreso académico en cada una de la materia, se podrá hacer la consulta de: Notas de Exámenes Parciales y finales, Notas de prácticas y seguimiento de competencias que van adquiriendo.
- Los docentes podrán hacer el registro y control del progreso académico de cada uno de los estudiantes inscritos en la materia de dicho docente
- Las notas de las Parciales deberán ser creadas de manera dinámica según la necesidad de la asignatura.
- Las notas de las Practicas deberán ser creadas de manera dinámica según la necesidad de la asignatura.
- Las Competencias deberán ser creadas de manera dinámica según la necesidad de la asignatura.

- Se deberá contar con un sistema de autenticación tanto para todos los usuarios Estudiante y Docentes.
- Creación Edición Eliminación y lectura de usuarios: Docentes y Estudiantes.

3.1.2 Requerimiento No Funcionales:

Se determino los siguientes requerimientos no funcionales:

- Solo tener acceso al sistema usuarios pertenecientes a la estudiante y docentes de la carrera de medina de la universidad pública de el alto.
- La aplicación debe estar disponible las 24 horas de día durante la gestión en curso.
- La aplicación debe requiere de una conexión a internet para la tener acceso a los módulos.
- La aplicación al estar desarrollado con tecnologías resientes es de fácil mantenibilidad y actualización así también con la posibilidad de agregar nuevas características.
- La aplicación esta desarrollada con un modelo de diseño responsive con lo cual se tendrá acceso desde cualquier dispositivo ya computadora Tablet o celular que tenga acceso a internet.

3.1.3 Clasificación de roles

Para la clasificación de los roles se toma en cuenta los usuarios que interactúan con el sistema de esta manera se organiza mejor el acceso a la información y así se tiene mejor control sobre los usuarios.

Tabla 3.1

Descripción de roles de usuario en el sistema

Rol	Descripción	Usuario

Administrador	Acceso a los módulos para Crear, Leer, Actualizar y Borrar los datos de docentes, estudiante, materias, competencias y calificaciones	Funcionario Administrativo de la carrera de medicina
Docente	Acceso a la creación de notas parciales y rellenado de campos de notas de estudiantes: practicas, parciales y competencias	Docentes de la carrera de medicina
Estudiante	Acceso a consultar los datos de sus competencias adquiridas así también notas de materias en curso (Parciales, Finales, Practicas)	Población Estudiantil de la carrera de medicina Universidad Carrera del Alto

3.1.4 Historias de usuario

A partir de los requerimientos funcionales descritos anteriormente, se elaboraron las siguientes historias de usuario las cuales se describen detalladamente a continuación.

Tabla 3.2

Historia de usuario de inicio de sesión o login

Titulo historia de usuario: Inicio de sesión	
Numero de historia de usuario: 1	
Usuario: Docente/ Estudiante/ Administrador	Prioridad: Media
Puntos de Historia: 2	Iteración: 1
Descripción: El usuario ingresa sus credenciales de acceso si son correctos accede a los módulos que son permitidos según su rol	

Observaciones: El usuario solo puede ingresar si sus credenciales son correctas en caso contrario solitaria la creación de una cuenta nueva al administrador del sistema

Tabla 3.3

Historia de usuario Registro Edición Eliminación y Lectura de Estudiantes

Título historia de usuario: Registro Edición Eliminación y Lectura de Estudiantes	
Numero de historia de usuario: 2	
Usuario: Administrador	Prioridad: Media
Puntos de Historia: 2	Iteración: 1
Descripción: El administrador registra edita elimina y lee los datos de los usuarios estudiantes	
Observación: Para la creación de un nuevo estudiante se requiere los siguientes datos personales del estudiante: Nombre completo y carnet de identidad, como datos opcionales correo dirección teléfono y fecha de nacimiento	

Tabla 3.4

Historia de usuario Registro Edición Eliminación y Lectura de Docentes

Titulo historia de usuario: Registro Edición Eliminación y Lectura de Docentes	
Numero de historia de usuario: 3	
Usuario: Administrador	Prioridad: Media

Puntos de Historia: 2 Iteración: 2
Descripción: El administrador registra edita elimina y lee los datos de los usuarios docentes
Observación: Observación: Para la creación de un nuevo Docente se requiere los siguientes datos personales del Docente: Nombre completo y carnet de identidad, como datos opcionales correo dirección teléfono y fecha de nacimiento

Tabla 3.5

Historia de usuario Registro Edición Eliminación y Lectura de Competencias

Titulo historia de usuario: Registro, Edición, Eliminación y Lectura de Competencias
Numero de historia de usuario: 4
Usuario: Administrador Prioridad: Media
Puntos de Historia: 2 Iteración: 2
Descripción: El administrador registra edita elimina y lee los datos de las competencias
Observación: Para la creación de una nueva competencia se requiere la descripción completa y así también el tipo de la competencia las cual esta definida en los tres tipos de competencias: Macro Competencias, Competencia Genérica, Competencia especifica.

Tabla 3.6

Historia de usuario Registro Edición Eliminación y Lectura de la Asignaturas

Titulo historia de usuario: Registro, Edición, Eliminación y Lectura de la Asignaturas
Numero de historia de usuario: 5
Usuario: Administrador Prioridad: Media
Puntos de Historia: 2 Iteración: 2
Descripción: El administrador registra edita elimina y lee los datos de las asignaturas
Observación: Para Asignar un docente a una asignatura dicho docente ya debe estar creado en el sistema, para la creación de la asignatura se solicitará los siguientes datos: Nombre de la Asignatura, sigla, carga horaria, numero de meses y el paralelo, para la asignación de competencias las dichas competencias ya deben estar creadas en el módulo de competencias.

Tabla 3.7

Historia de usuario Registro Edición Eliminación y Lectura de Calificaciones

Titulo historia de usuario: Creación, Eliminación y Lectura de Calificaciones
Numero de historia de usuario: 6
Usuario: Administrador Prioridad: Media
Puntos de Historia: 2 Iteración: 3
Descripción: El administrador elimina y lee los datos de las calificaciones de cada uno de los estudiantes en cada asignatura cursada ya sea calificaciones parciales, finales y/o practicas

Observación: Por motivos de seguridad el docente es el único que puede crear y editar las calificaciones de cada estudiante

Tabla 3.8

Historia de usuario Creación y Registro de Calificaciones de Parciales y Finales

Titulo historia de usuario: Creación y Registro de Calificaciones de Parciales	
Numero de historia de usuario: 7	
Usuario: Docente	Prioridad: Media
Puntos de Historia: 2	Iteración: 3
Descripción: El docente puede crear campos para las notas de los parciales según sea necesario en la asignatura que desarrolla así también llenar los registros de dichas calificaciones para llevar el control de estudiante	
Observación: Ninguna	

Tabla 3.9

Historia de usuario Registro de Calificaciones de Practicas

Titulo historia de usuario: Creación y registro de Calificaciones de Practicas	
Numero de historia de usuario: 8	
Usuario: Docente	Prioridad: Media
Puntos de Historia: 2	Iteración 3
Descripción: El docente llena los campos de acuerdo hoja de calificación personal de prácticas designada de cada asignatura	
Observación: Ninguna	

Tabla 3.10

Historia de usuario Registro de Competencias

Titulo historia de usuario: Asignación de competencias a estudiante	
Numero de historia de usuario: 9	
Usuario: Docente	Prioridad: Media
Puntos de Historia: 2	Iteración: 3
Descripción: El docente realiza el registro sobre la competencia cumplida por el estudiante en su materia dependiendo de los criterios propios de cada asignatura.	
Observación: las competencias listadas solo aparecerán si han sido designadas en la asignatura por el administrador.	

Tabla 3.11

Historia de usuario Lectura de Calificaciones y Competencias

Titulo historia de usuario: Lectura de Calificaciones y Competencias	
Numero de historia de usuario: 10	
Usuario: Estudiante	Prioridad: Media
Puntos de Historia: 2	Iteración 4
Descripción: El estudiante puede observar las calificaciones y competencias obtenidas que ya se hayan registrado por el docente en la asignatura que desarrolla.	
Observación: las calificaciones de prácticas o parciales solo serán listadas si el docente ya realizo la calificación de parcial o prácticas, el mismo caso con las competencias solo aparecerán si han sido asignadas al estudiante.	

Tabla 3.12

Historia de usuario Cerrar sesión

Titulo historia de usuario: Cerrar sesión
Numero de historia de usuario: 11
Usuario: Administrador/Docente/Estudiante — Prioridad: Media
Puntos de Historia: 2 — Iteración 4
Descripción: Al finalizar las consultas deseados los usuarios cierran la sesión activa y siendo redireccionados a la página de inicio de sesión
Observación: Al cerrar la sesión se destruye el token en el dispositivo el cual esta activa la sesión

Tabla 3.13

Historia de usuario Inscripción de Estudiantes

Titulo historia de usuario: Inscripción de Estudiantes
Numero de historia de usuario: 12
Usuario: Administrador — Prioridad: Media
Puntos de Historia: 2 — Iteración: 4
Descripción: El administrador registra las asignaturas en las cuales están inscritos los estudiantes en la gestión en curso, así también se puede eliminar la inscripción de dicho estudiante
Observación: Para la inscripción del estudiante la asignatura debe ser creada con anterioridad así mismo la cuenta del estudiante debe haber sido creada con anterioridad para que aparezca listada en el módulo de inscripción

Luego de haber realizado las historias de los casos de uso, se elaboro un listado general el cual se describe en la siguiente tabla

Tabla 3.14

Lista de Historias de usuario

Numero de historia de usuario	Título de la historia de usuario
1	Inicio de sesión
2	Registro Edición Eliminación y Lectura de Estudiantes
3	Registro Edición Eliminación y Lectura de Docentes
4	Registro, Edición, Eliminación y Lectura de Competencias
5	Registro, Edición, Eliminación y Lectura de la Asignaturas
6	Eliminación y Lectura de Calificaciones
7	Creación y Registro de Calificaciones de Parciales y Finales
8	Creación y registro de Calificaciones de Practicas
9	Asignación de competencias a estudiante
10	Lectura de Calificaciones y Competencias
11	Cerrar sesión
12	Inscripción de Estudiantes

3.1.5 Plan de publicación

Con las historias de usuario ya definidas se procede a crear el plan de trabajo, con el cual se realiza un plan para cumplir con el desarrollo en un tiempo óptimo para la entrega de sistema

Tabla 3.15

Tabla de cronograma de publicación de iteraciones

Iteración	Historia	Abril				Mayo				Junio			
		1	2	3	4	1	2	3	4	1	2	3	4
Primera	1,2												
Segunda	3,4,5,6												
Tercera	7,8,9,												
Cuarta	10,11,12												

3.1.6 Iteraciones

En el plan de publicación de iteraciones de la figura 3.15 se designan fechas no mayores a 4 semanas para el desarrollo, ahora se procederá a dividir en cada iteración en tareas cada uno de las historias de usuario

3.1.6.1 Primera Iteración

Las historias de usuario que se desarrollaron en esta primera iteración son:

Tabla 3.16

Historias de usuario de la primera iteración

Numero de historia de usuario	Título de historia de usuario
1	Inicio de sesión
2	Registro Edición Eliminación y Lectura de Estudiantes

En la tabla 3.17 se elabora la designación de las tareas de la primera iteración.

Tabla 3.17

Tareas de Primera iteración

Numero de Tarea	Numero de historia de usuario	Nombre de la tarea
1	1	Inicialización de proyecto Backend y Frontend
2	1	Creación de módulo Login en el backend
3	1	Diseño y desarrollo de la interfaz de usuario del Login en frontend
4	2	Creación de la tabla de estudiantes en la base de datos
5	2	Creación del módulo de Estudiantes en el backend
6	2	Diseño y desarrollo de la interfaz del módulo de estudiantes en el frontend

3.1.6.1.1 Desarrollo de tareas de la primera iteración

Se elaboran las tarjetas de tareas en función a las tareas descritas en la Tabla 3.17.

Tabla 3.18

Tarjeta de Tarea n°1

Numero de Tarea: 1	Numero de historia de usuario: 1
Nombre de la tarea: Inicialización de proyecto Backend y Frontend	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 2 Días

Programador Responsable: Otmar Kevin Aguilar Leon

Descripción: Se realizo la inicializa de los ambientes de desarrollo tanto frontend como backend.

Backend: El cual está realizado con el framework Nestjs, como orm para el manejo de la base de datos se usa la biblioteca Typeorm con la cual se usa la conexión a la base de datos MariaDB y se inicializa el repositorio Git para el manejo de versiones adecuado.

Frontend: Se inicializa el proyecto Angular y se realiza la instalación de los paquetes necesarios para el desarrollo de la aplicación, se realiza la importación PrimeNG la cual es la biblioteca de componentes que se usó en el proyecto y se inicializa el repositorio Git para el manejo de versiones adecuado.

Tabla 3.19

Tarjeta de Tarea n°2

Numero de Tarea: 2	Numero de historia de usuario: 1

Nombre de la tarea: Creación de módulo Login en el backend

Tipo de tarea: Desarrollo	Tiempo de desarrollo: 4 Días

Programador Responsable: Otmar Kevin Aguilar Leon

Descripción: Se define el uso de la herramienta JWT para el manejo de la seguridad y roles de la aplicación, Se crea los controladores y servicios para el manejo de los endpoints del módulo de login.

Tabla 3.20

Tarjeta de Tarea n°3

Numero de Tarea: 3	Numero de historia de usuario: 1

Nombre de la tarea: Diseño y desarrollo de la interfaz de usuario del Login en frontend

Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Se crea en módulo de login y sus respectivos servicios para la comunicación con el frontend, Se da un estilo al módulo de login con una interfaz sencilla y eficaz	

Tabla 3.21

Tarjeta de Tarea n°4

Numero de Tarea: 4	Numero de historia de usuario: 2
Nombre de la tarea: Creación de la tabla de estudiantes en la base de datos	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Se declara los entites para la creación de la tabla de estudiantes en la base de datos la cual está relacionada con la tabla de usuarios	

Tabla 3.22

Tarjeta de Tarea n°5

Numero de Tarea: 5	Numero de historia de usuario: 2
Nombre de la tarea: Creación del módulo de Estudiantes en el backend	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Se realiza la creación de los controladores y los servicios y se crean los métodos para el manejo de las instancias del repositorio de estudiantes, se crean los endpoints para el listado, creación y eliminación de estudiantes.	

Tabla 3.23

Tarjeta de Tarea n°6

Numero de Tarea: 6	Numero de historia de usuario: 2
Nombre de la tarea: Diseño y desarrollo de la interfaz del módulo de estudiantes en el frontend	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Creación de módulo de estudiantes en el cual se realiza la gestión del listado, creación, edición y eliminación de los estudiantes en cual se comunica con los endpoints del módulo estudiantes del backend	

3.1.6.2 Segunda Iteración

Las historias de usuario que se desarrollaron en esta segunda iteración son:

Tabla 3.24

Historias de usuario de la segunda iteración

Numero de historia de usuario	Título de historia de usuario
3	Registro Edición Eliminación y Lectura de Docentes
4	Registro, Edición, Eliminación y Lectura de Competencias
5	Registro, Edición, Eliminación y Lectura de la Asignaturas
6	Creación, Eliminación y Lectura de Calificaciones

En la tabla 3.25 se elabora la designación de las tareas de la segunda iteración.

Tabla 3.25

Tareas de Segunda iteración

Numero de Tarea	Numero de historia de usuario	Nombre de la tarea
7	3	Creación de la tabla de docentes en la base datos
8	3	Creación del módulo de Docentes en el backend
9	3	Diseño y desarrollo de la interfaz del módulo de docentes en el frontend
10	4	Creación de la tabla de competencias en la base de datos
11	4	Creación del módulo de competencias en el backend
12	4	Diseño y desarrollo de la interfaz del módulo de competencias en el frontend
13	5	Creación de la tabla de asignaturas en la base de datos
14	5	Creación del módulo de asignaturas en el backend
15	5	Diseño y desarrollo de la interfaz del módulo de asignaturas en el frontend

16	5	Asignación de competencias a asignaturas(backend-frontend), creación de la relación en la base de datos
17	6	Creación de la tabla de calificaciones en la base de datos
18	6	Creación del módulo de calificaciones en el backend
19	6	Diseño y desarrollo de la interfaz del módulo de calificaciones en el frontend

3.1.6.2.1 Desarrollo de tareas de la Segunda Iteración

Se elaboran las tarjetas de tareas en función a las tareas descritas en la Tabla 3.25.

Tabla 3.26

Tarjeta de Tarea n°7

Numero de Tarea: 7	Numero de historia de usuario: 3
Nombre de la tarea: Creación de la tabla de docentes en la base datos	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Se declara los entites para la creación de la tabla de docentes en la base de datos la cual está relacionada con la tabla de usuarios	

Tabla 3.27

Tarjeta de Tarea n°8

Numero de Tarea: 8	Numero de historia de usuario: 3
Nombre de la tarea: Creación del módulo de Docentes en el backend	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Se realiza la creación de los controladores y los servicios y se crean los métodos para el manejo de las instancias del repositorio de docentes, se crean los endpoints para la listado, creación y eliminación de docentes.	

Tabla 3.28

Tarjeta de Tarea n°9

Numero de Tarea: 9	Numero de historia de usuario: 3
Nombre de la tarea: Diseño y desarrollo de la interfaz del módulo de docentes en el frontend	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Creación de módulo de estudiantes en el cual se realiza la gestión del listado, creación, edición y eliminación de los estudiantes en cual se comunica con los endpoints del módulo docentes del backend	

Tabla 3.29

Tarjeta de Tarea n°10

Numero de Tarea: 10	Numero de historia de usuario: 4

Nombre de la tarea: Creación de la tabla de competencias en la base de datos
Tipo de tarea: Desarrollo Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon
Descripción: Se declara los entites para la creación de la tabla de competencias en la base de datos.

Tabla 3.30

Tarjeta de Tarea n°11

Numero de Tarea: 11 Numero de historia de usuario: 4
Nombre de la tarea: Creación del módulo de competencias en el backend
Tipo de tarea: Desarrollo Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon
Descripción: Se realiza la creación de los controladores y los servicios y se crean los métodos para el manejo de las instancias del repositorio de competencias, se crean los endpoints para la listado, creación y eliminación de competencias.

Tabla 3.31

Tarjeta de Tarea n°12

Numero de Tarea: 12 Numero de historia de usuario: 4
Nombre de la tarea: Diseño y desarrollo de la interfaz del módulo de competencias en el frontend
Tipo de tarea: Desarrollo Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon

Descripción: Creación de módulo de competencias en el cual se realiza la gestión del listado, creación, edición y eliminación de las competencias en cual se comunica con los endpoints del módulo competencias del backend

Tabla 3.32

Tarjeta de Tarea n°13

Numero de Tarea: 13	Numero de historia de usuario: 5
Nombre de la tarea: Creación de la tabla de asignaturas en la base de datos	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Se declara los entites para la creación de la tabla de asignaturas en la base de datos.	

Tabla 3.33

Tarjeta de Tarea n°14

Numero de Tarea: 14	Numero de historia de usuario: 5
Nombre de la tarea: Creación del módulo de asignaturas en el backend	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Se realiza la creación de los controladores y los servicios y se crean los métodos para el manejo de las instancias del repositorio de asignaturas, se crean los endpoints para la listado, creación y eliminación de asignaturas.	

Tabla 3.34

Tarjeta de Tarea n°15

Numero de Tarea: 15	Numero de historia de usuario: 5
Nombre de la tarea: Diseño y desarrollo de la interfaz del módulo de asignaturas en el frontend	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Creación de módulo de competencias en el cual se realiza la gestión del listado, creación, edición y eliminación de las asignaturas en cual se comunica con los endpoints del módulo asignaturas del backend	

Tabla 3.35

Tarjeta de Tarea n°16

Numero de Tarea: 16	Numero de historia de usuario: 5
Nombre de la tarea: Asignación de competencias a asignaturas(backend-frontend), creación de la relación en la base de datos	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: En el backend se realiza una tabla pivote donde se relacionan la tabla de asignaturas y competencias, así también se crean los controladores y servicios para la creación de los endpoints del módulo de asignaturas-competencias, En el frontend se desarrolla una interfaz con la cual se puede relacionar las competencias con las asignaturas de manera sencilla e intuitiva.	

Tabla 3.36

Tarjeta de Tarea n°17

Numero de Tarea: 17	Numero de historia de usuario: 6
Nombre de la tarea: Creación de la tabla de calificaciones en la base de datos	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Se declara los entites para la creación de la tabla de calificaciones en la base de datos.	

Tabla 3.37

Tarjeta de Tarea n°18

Numero de Tarea: 18	Numero de historia de usuario: 6
Nombre de la tarea: Creación del módulo de calificaciones en el backend	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Se realiza la creación de los controladores y los servicios y se crean los métodos para el manejo de las instancias del repositorio de calificaciones, se crean los endpoints para la listado, creación y eliminación de calificaciones.	

Tabla 3.38

Tarjeta de Tarea n°19

Numero de Tarea: 19	Numero de historia de usuario: 6
Nombre de la tarea: Diseño y desarrollo de la interfaz del módulo de calificaciones en el frontend	

Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Creación de módulo de competencias en el cual se realiza la gestión del listado, creación y eliminación de las calificaciones en cual se comunica con los endpoints del módulo calificaciones en el backend	

3.1.6.3 Tercera Iteración

Las historias de usuario que se desarrollaron en esta tercera iteración son:

Tabla 3.39

Historias de usuario de la tercera iteración

Numero de historia de usuario	Título de historia de usuario
7	Creación y Registro de Calificaciones de Parciales
8	Creación y registro de Calificaciones de Practicas
9	Asignación de competencias a estudiante

En la tabla 3.40 se elabora la designación de las tareas de la segunda iteración.

Tabla 3.40

Tareas de tercera iteración

Numero de Tarea	Numero de historia de usuario	Nombre de la tarea
20	7	Creación de la tabla de relación entre calificación y estudiantes

21	7	Creación de la tabla de relación entre prácticas y estudiantes
22	7	Creación de los endpoints para el manejo de las calificaciones del estudiante
23	8	Creación de los endpoints para el manejo de las practicas del estudiante
24	7	Creación de la interfaz para la calificación de parciales de estudiantes
25	8	Creación de la interfaz para la calificación de Practicas de estudiantes
26	9	Creación de la tabla de relación entre competencia y estudiante
27	9	Creación de los endpoints para el manejo de las competencias del estudiante
28	9	Creación de la interfaz para la adjudicación de competencias a estudiante

3.1.6.3.1 Desarrollo de tareas Tercera Iteración

Se elaboran las tarjetas de tareas en función a las tareas descritas en la Tabla 3.40.

Tabla 3.41

Tarjeta de Tarea n°20

Numero de Tarea: 20	Numero de historia de usuario: 7
Nombre de la tarea: Creación de la tabla de relación entre calificación y estudiantes	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Se crea la relación entre las tablas de calificación y estudiantes en una tabla pivote, en cual se lleva el registro de las calificaciones de los estudiantes.	

Tabla 3.42

Tarjeta de Tarea n°21

Numero de Tarea: 21	Numero de historia de usuario: 7
Nombre de la tarea: Creación de la tabla de relación entre prácticas y estudiantes	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Se crea la relación entre las tablas de prácticas y estudiantes en una tabla pivote, en cual se lleva el registro de las calificaciones de las prácticas de los estudiantes.	

Tabla 3.43

Tarjeta de Tarea n°22

Numero de Tarea: 22	Numero de historia de usuario: 7
Nombre de la tarea: Creación de los endpoints para el manejo de las calificaciones del estudiante	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días

Programador Responsable: Otmar Kevin Aguilar Leon
Descripción: Creación de controladores y servicios para el manejo de las calificaciones de los estudiantes

Tabla 3.44

Tarjeta de Tarea n°23

Numero de Tarea: 23	Numero de historia de usuario: 8
Nombre de la tarea: Creación de los endpoints para el manejo de las practicas del estudiante	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Creación de controladores y servicios para el manejo de las calificaciones de las prácticas de los estudiantes	

Tabla 3.45

Tarjeta de Tarea n°24

Numero de Tarea: 24	Numero de historia de usuario: 7
Nombre de la tarea: Creación de la interfaz para la calificación de parciales de estudiantes	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Creación del módulo de calificación de la asignatura para calificación de parciales de cada uno de los estudiantes.	

Tabla 3.46

Tarjeta de Tarea n°25

Numero de Tarea: 25	Numero de historia de usuario: 8
Nombre de la tarea: Creación de la interfaz para la calificación de Practicas de estudiantes	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Creación del módulo de calificación de la asignatura para calificación de prácticas de cada uno de los estudiantes.	

Tabla 3.47

Tarjeta de Tarea n°26

Numero de Tarea: 26	Numero de historia de usuario: 9
Nombre de la tarea: Creación de la tabla de relación entre competencia y estudiante	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Se crea la tabla pivote donde se guarda las competencias de cada estudiante, las competencias que ya le fueron designadas	

Tabla 3.48

Tarjeta de Tarea n°27

Numero de Tarea: 27	Numero de historia de usuario: 9
Nombre de la tarea: Creación de los endpoints para el manejo de las competencias del estudiante	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días

Programador Responsable: Otmar Kevin Aguilar Leon
Descripción: Creación de los servicios y controladores para la asociación de competencias a cada estudiante

Tabla 3.49

Tarjeta de Tarea n°28

Numero de Tarea: 28	Numero de historia de usuario: 9
Nombre de la tarea: Creación de la interfaz para la adjudicación de competencias a estudiante	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Creación de una interfaz sencilla e intuitiva para la asignación de competencias a cada estudiante.	

3.1.6.4 Cuarta Iteración

Las historias de usuario que se desarrollaron en esta cuarta iteración son:

Tabla 3.50

Historias de usuario de la cuarta iteración

Numero de historia de usuario	Título de historia de usuario
10	Lectura de Calificaciones y Competencias
11	Cerrar Sesión
12	Inscripciones de Estudiantes

En la tabla 3.51 se elabora la designación de las tareas de la cuarta iteración.

Tabla 3.51

Tareas de Cuarta iteración

Numero de Tarea	Numero de historia de usuario	Nombre de la tarea
29	10	Creación de endpoints para listar calificaciones y competencias de estudiantes
30	10	creación de interfaz para el listado de las calificaciones y competencias de los estudiantes
31	11	Eliminación del token y redirigir al login
32	12	creación de la tabla de inscripciones
33	12	creación de endpoints de inscripción
34	12	Creación de la Interfaz de inscripción

3.1.6.4.1 Desarrollo de tareas Cuarte Iteración

Se elaboran las tarjetas de tareas en función a las tareas descritas en la Tabla 3.51.

Tabla 3.52

Tarjeta de Tarea n°29

Numero de Tarea: 29	Numero de historia de usuario: 10

Nombre de la tarea: Creación de endpoints para listar calificaciones y competencias de estudiantes
Tipo de tarea: Desarrollo Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon
Descripción: creación de servicios y controladores para listar las calificaciones y competencias adquiridas de cada uno de los estudiantes

Tabla 3.53

Tarjeta de Tarea n°30

Numero de Tarea: 30 Numero de historia de usuario: 10
Nombre de la tarea: Creación de interfaz para el listado de las calificaciones y competencias de los estudiantes
Tipo de tarea: Desarrollo Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon
Descripción: Se crea una interfaz donde se pueda listas las calificaciones obtenidas por el estudiante donde se puede observar los la calificación obtenida, la descripción de la calificación y la asignatura a la cual pertenece la calificación, de igual manera con las competencias se lista las competencias adquiridas por el estudiante y se observan datos como descripción de la competencia, tipo de competencias y asignatura en la cual se obtuvo la competencia.

Tabla 3.54

Tarjeta de Tarea n°31

Numero de Tarea: 31 Numero de historia de usuario: 11
Nombre de la tarea: Eliminación del token y redirigir al login

Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Se destruye el token jwt de la sesión y se redirecciona a la página de login para que se inicie una nueva sesión	

Tabla 3.55

Tarjeta de Tarea n°32

Numero de Tarea: 32	Numero de historia de usuario: 12
Nombre de la tarea: Creación de la tabla de inscripciones	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Se crea la tabla de inscripciones donde se almacena los estudiantes inscritos en cada asignatura y así también se registra la gestión.	

Tabla 3.56

Tarjeta de Tarea n°33

Numero de Tarea: 33	Numero de historia de usuario: 12
Nombre de la tarea: Creación de endpoints de inscripción	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Creación de servicio y controladores para el registro de la inscripción de los estudiantes.	

Tabla 3.57

Tarjeta de Tarea n°34

Numero de Tarea: 34	Numero de historia de usuario: 12
Nombre de la tarea: Creación de la Interfaz de inscripción	
Tipo de tarea: Desarrollo	Tiempo de desarrollo: 5 Días
Programador Responsable: Otmar Kevin Aguilar Leon	
Descripción: Se crea una interfaz sencilla e intuitiva para la inscripción de los estudiantes en las asignaturas que le corresponde.	

3.2 Fase de Diseño

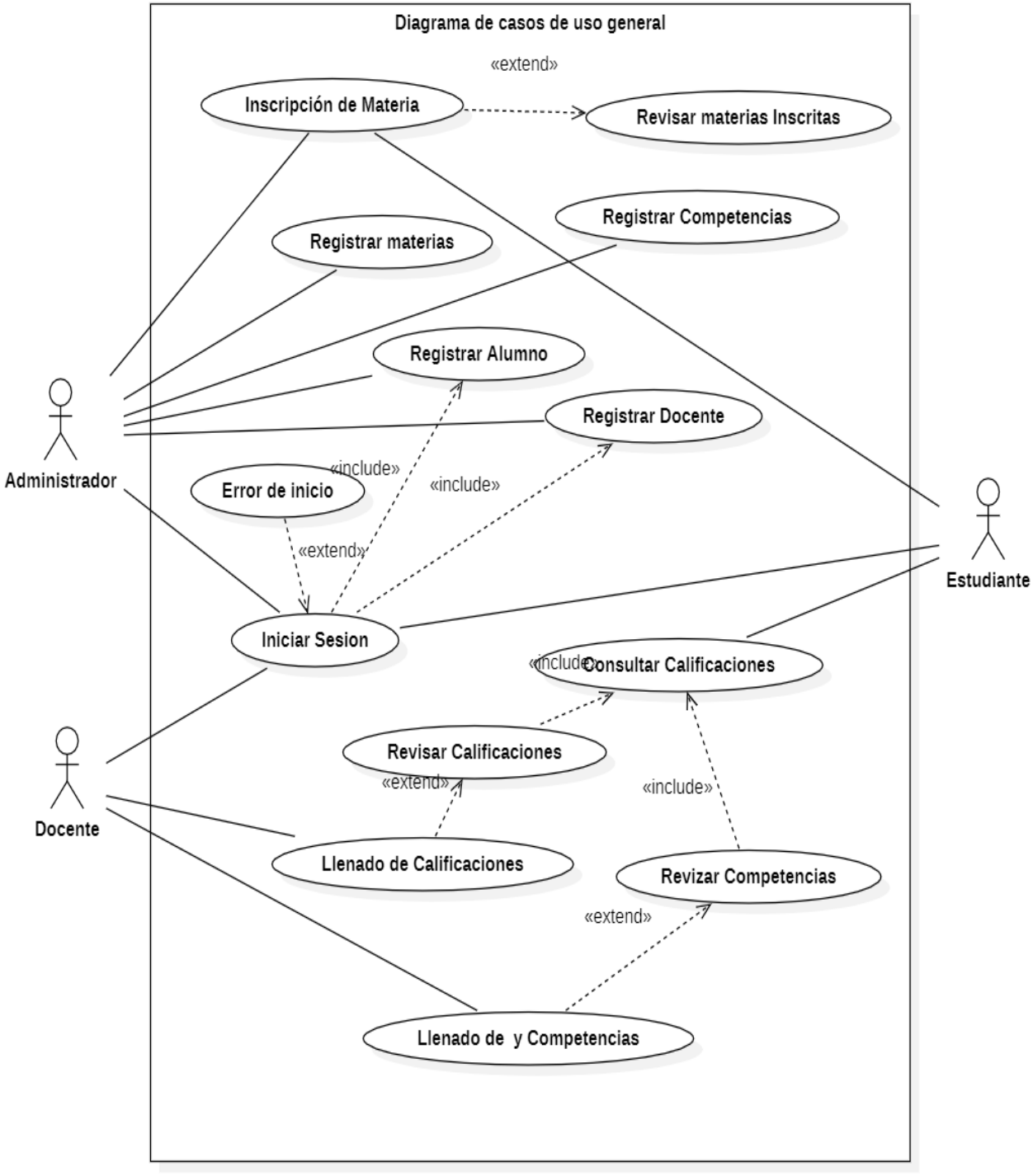
La segunda fase de la metodología XP, se presentarán diseños simples y precisos para lograr una mejor comprensión de la funcionalidad del sistema en su conjunto esto logrará desarrollar en menos tiempo.

3.2.1 Diagrama general de casos de uso

Este modelo se presenta mediante diagramas de Casos de Uso el cual se piensan en cuáles serán las principales funcionalidades que el software debe permitir llevar a cabo, mientras que los actores representan los roles o papeles que actúan recíprocamente con los procesos.

Figura 3.1

Diagrama de Casos de Uso General



3.2.2 Diagramas de casos de uso específicos

Los diagramas a continuación las situaciones más recurrentes en el sistema

Figura 3.2

Diagrama de caso de uso: inicio y cierre de sesión

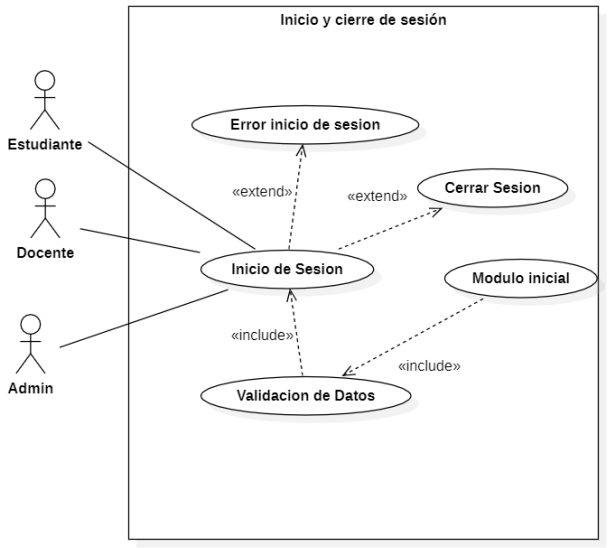


Figura 3.3

Diagrama de casos de uso: Gestión de usuarios

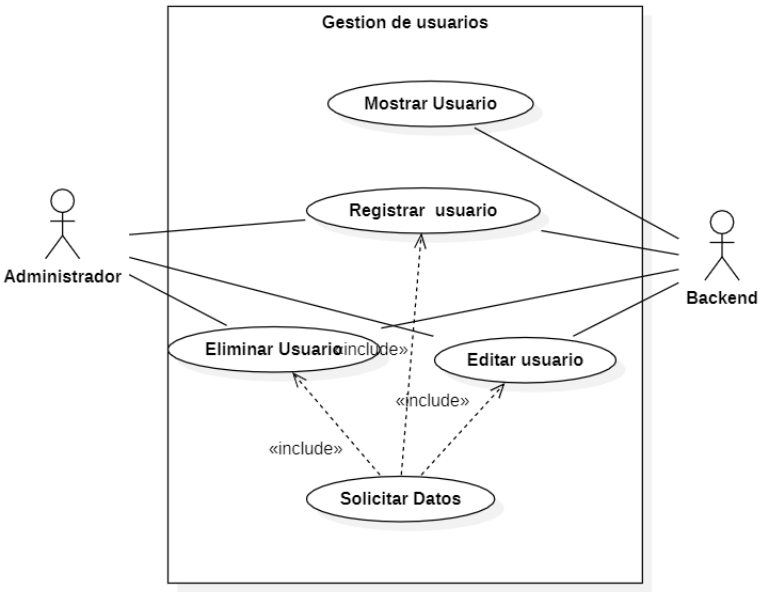


Figura 3.4

Diagrama de caso de uso: Administración de parciales y prácticas

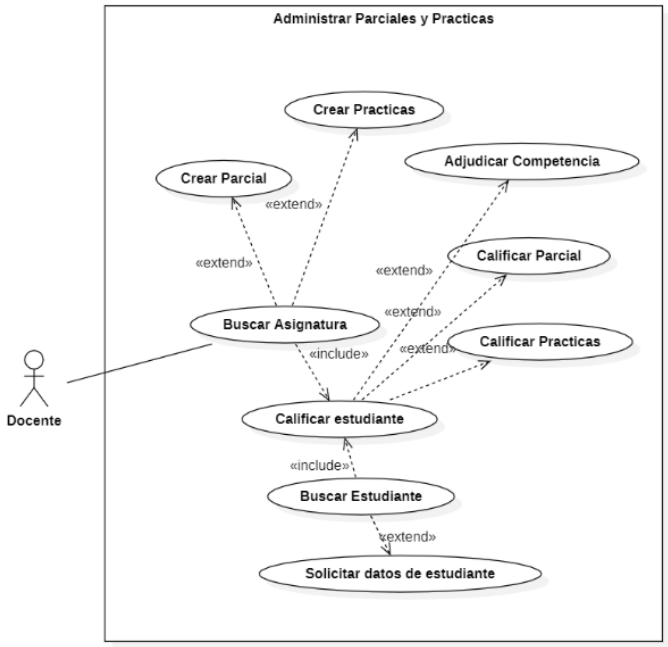


Figura 3.5

Diagrama de caso de uso: Revisar Calificaciones y competencias

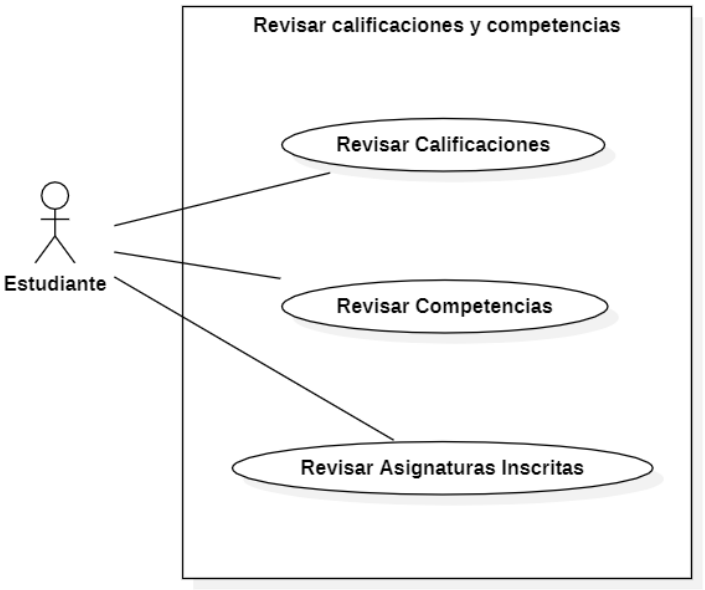


Figura 3.6

Diagrama de caso de uso: inscripción de estudiante

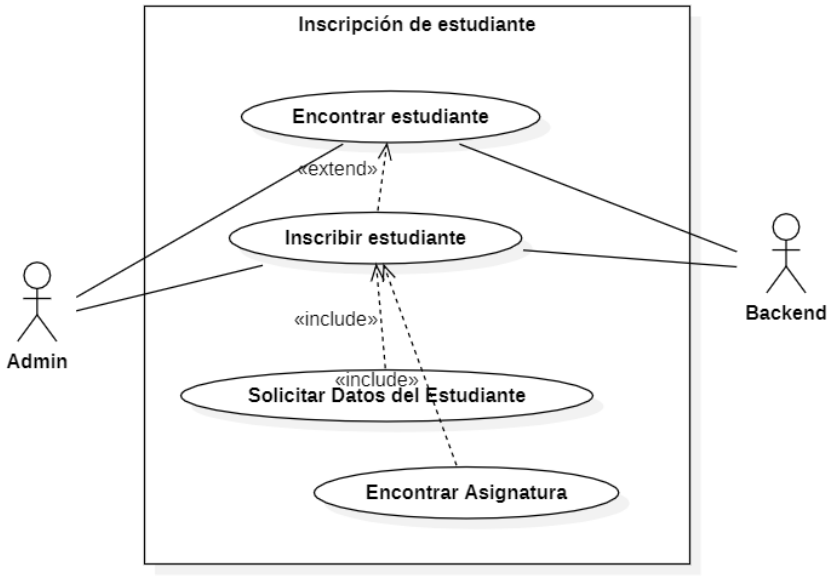


Figura 3.7

Diagrama de caso de uso: Gestión de competencias

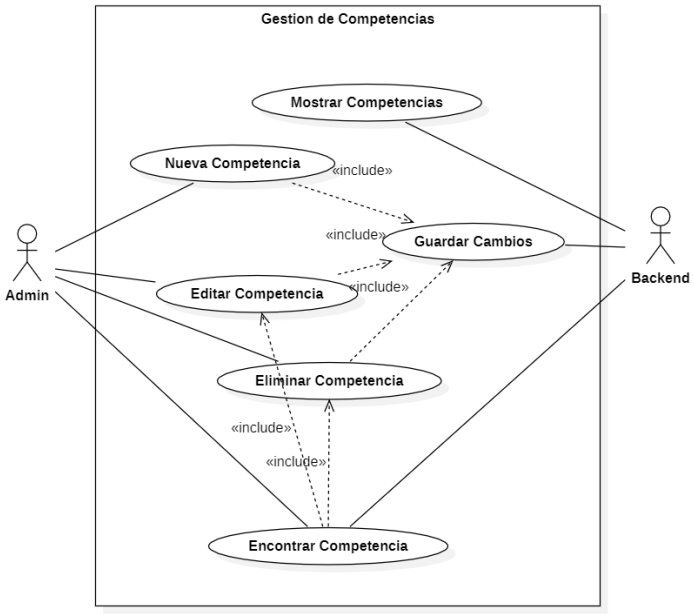
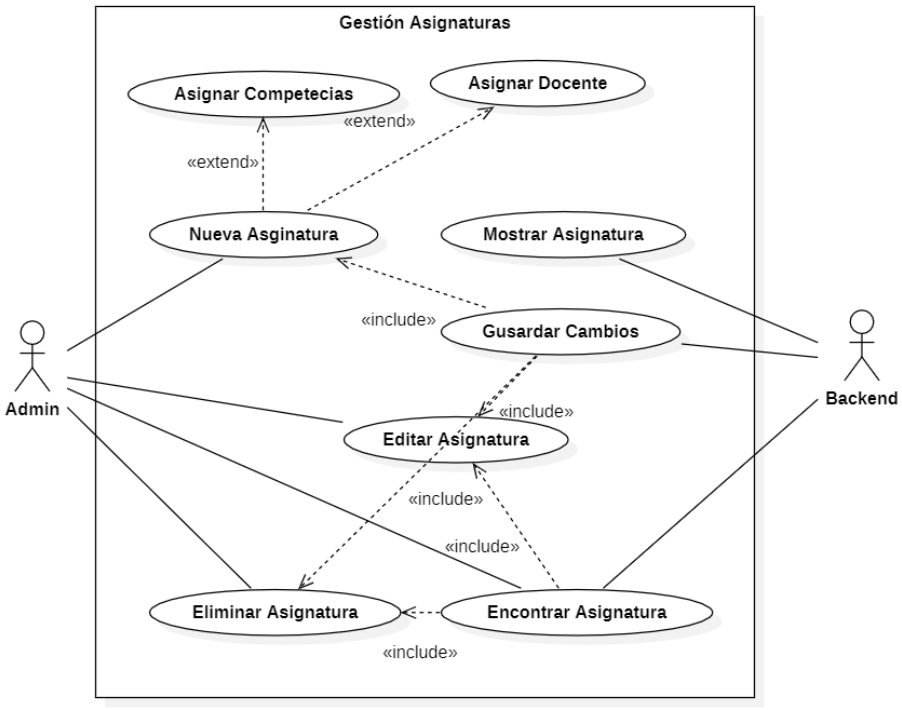


Figura 3.8

Diagrama de caso de uso: Gestión de asignaturas



El diagrama de casos de uso es fundamental para la realización del diagrama entidad-relación, en este caso todas las clases pasan como entidades y los atributos de las clases como atributos de la entidad

3.2.3 Diagramas de Secuencia

Figura 3.9

Diagrama de secuencia: Administración de usuario

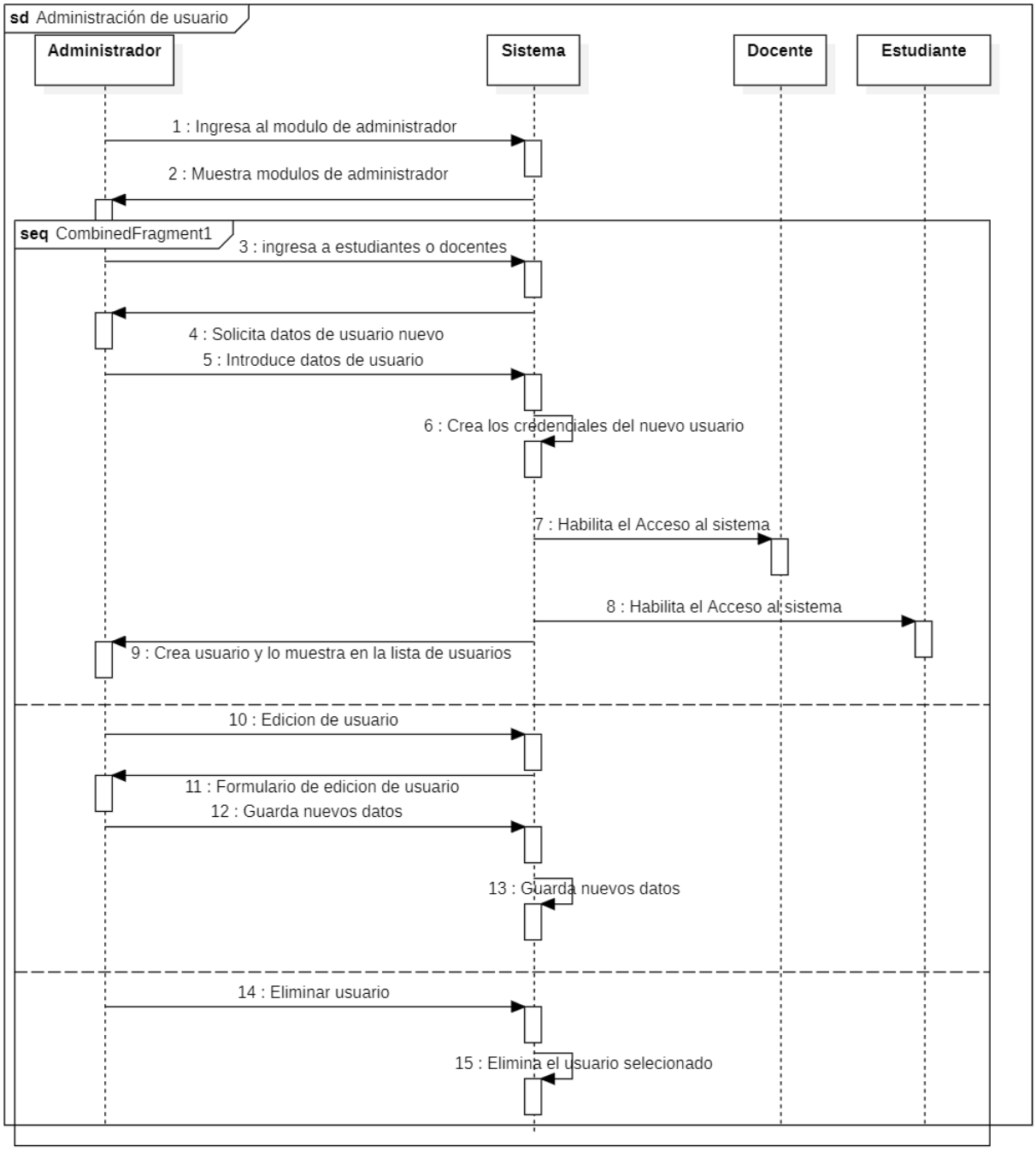


Figura 3.10

Diagrama de secuencia: Login

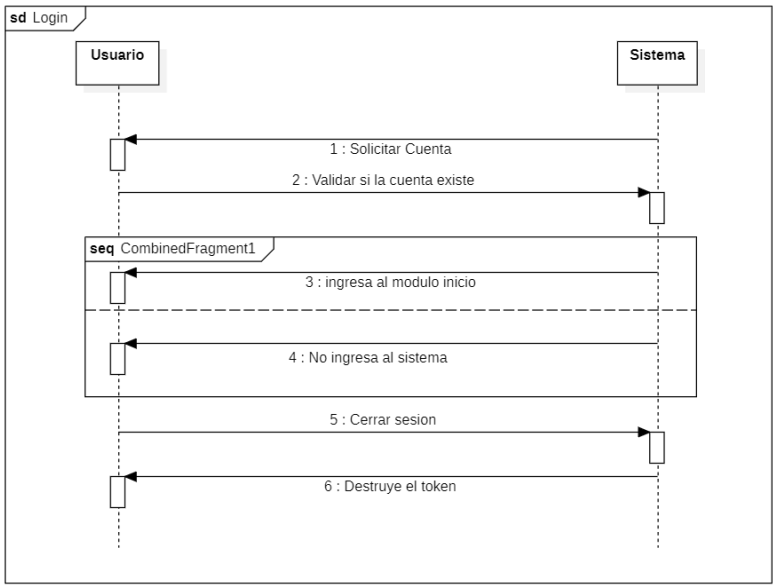


Figura 3.11

Diagrama de secuencia: Inscripción

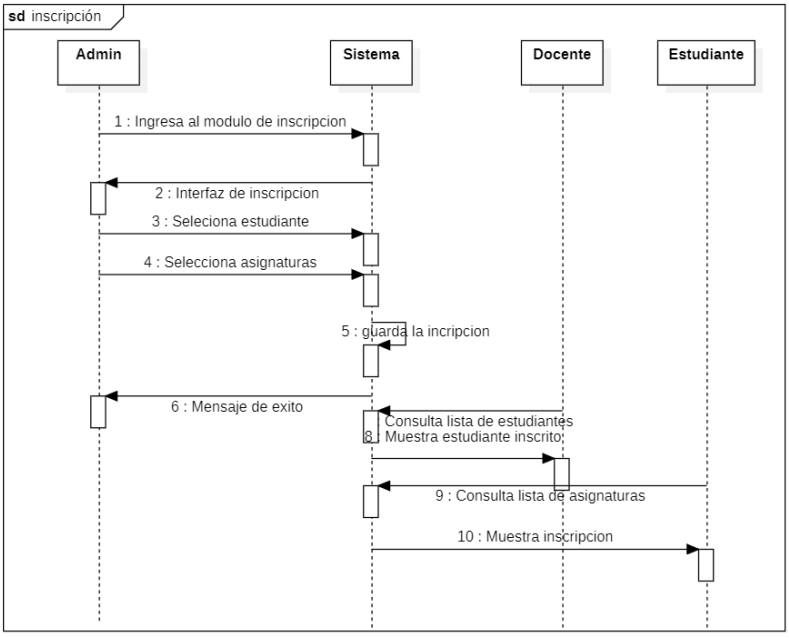


Figura 3.12

Diagrama de secuencia: Administración de calificaciones y competencias

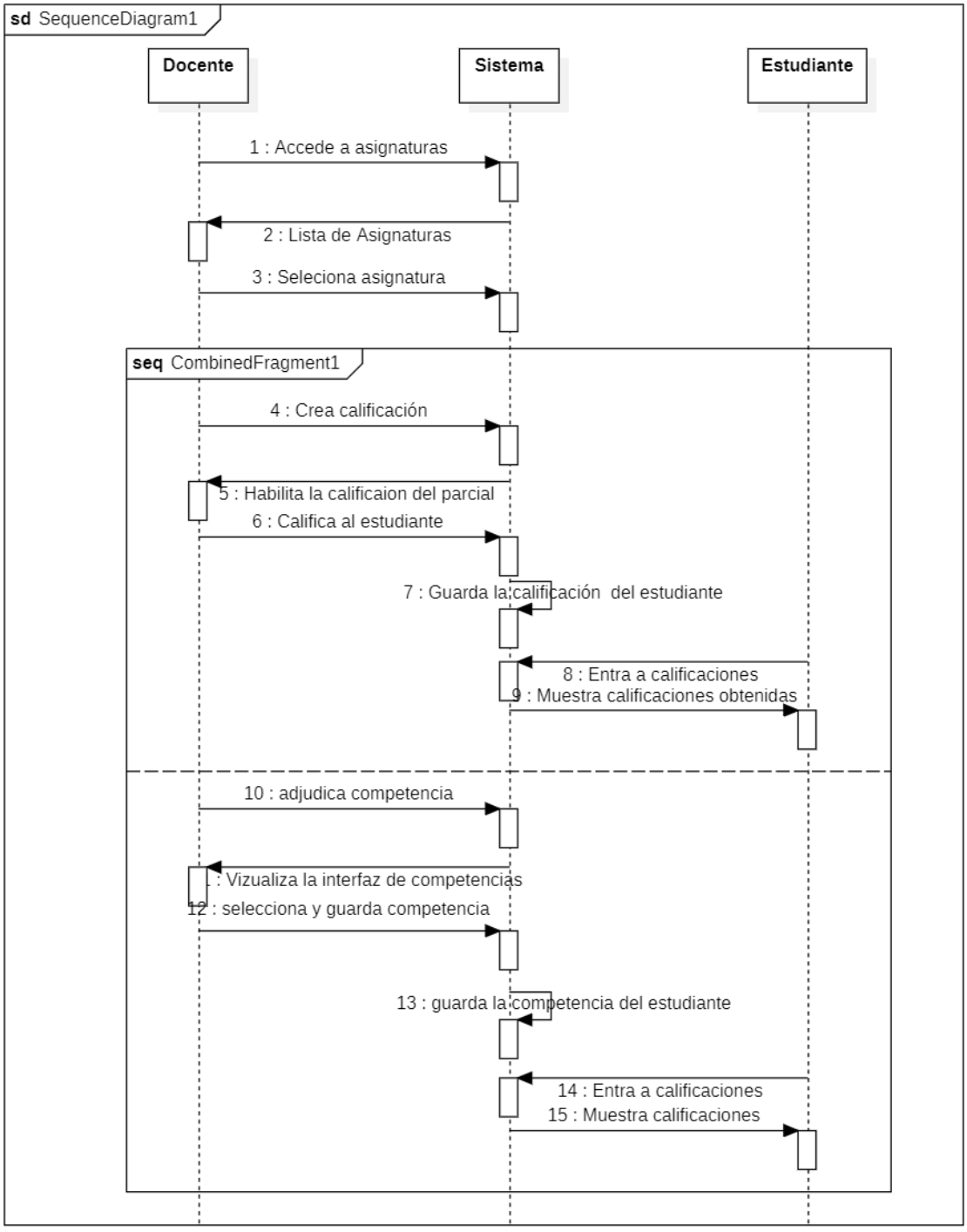


Figura 3.13

Diagrama de secuencia: Administración de asignaturas

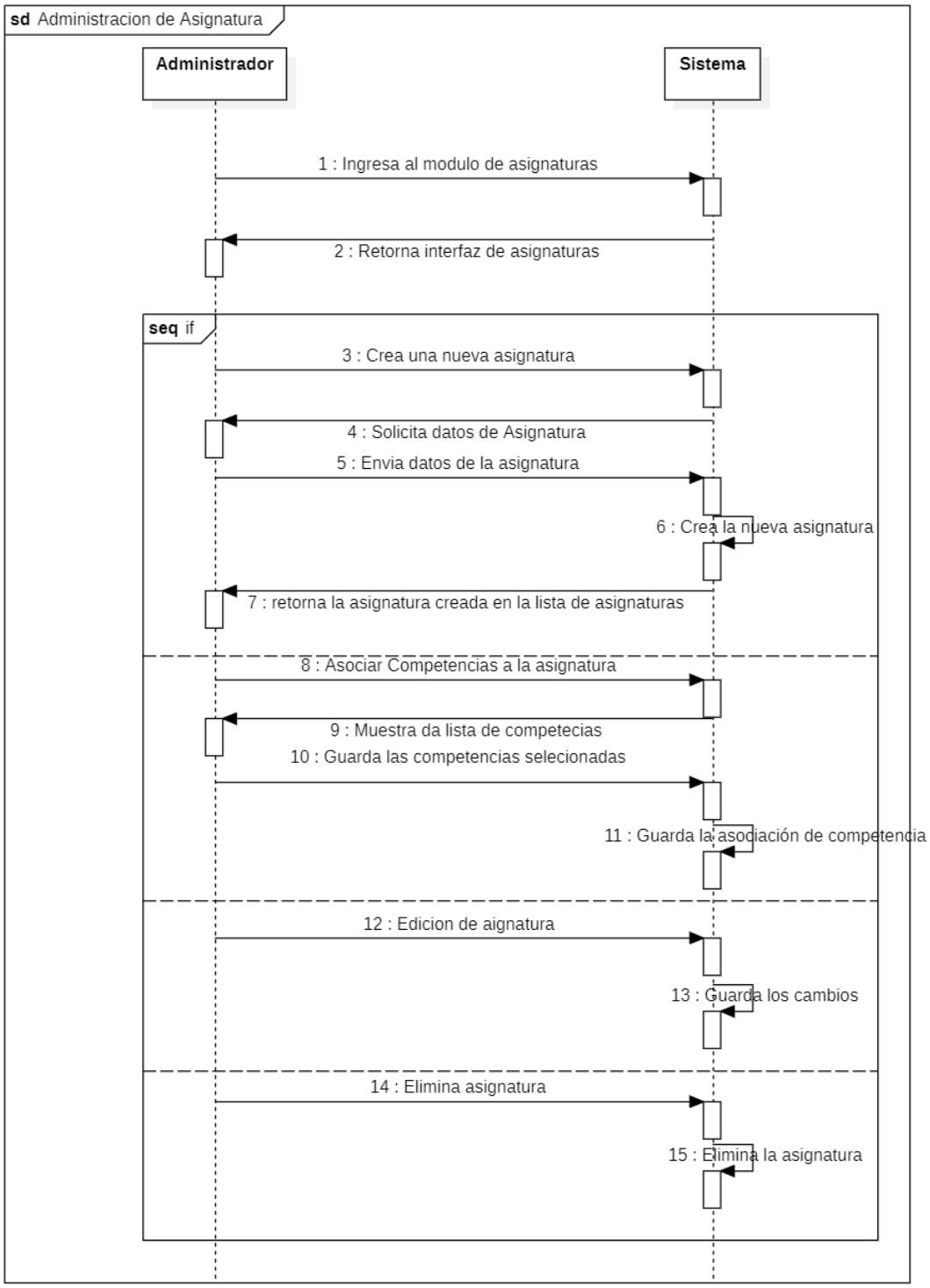
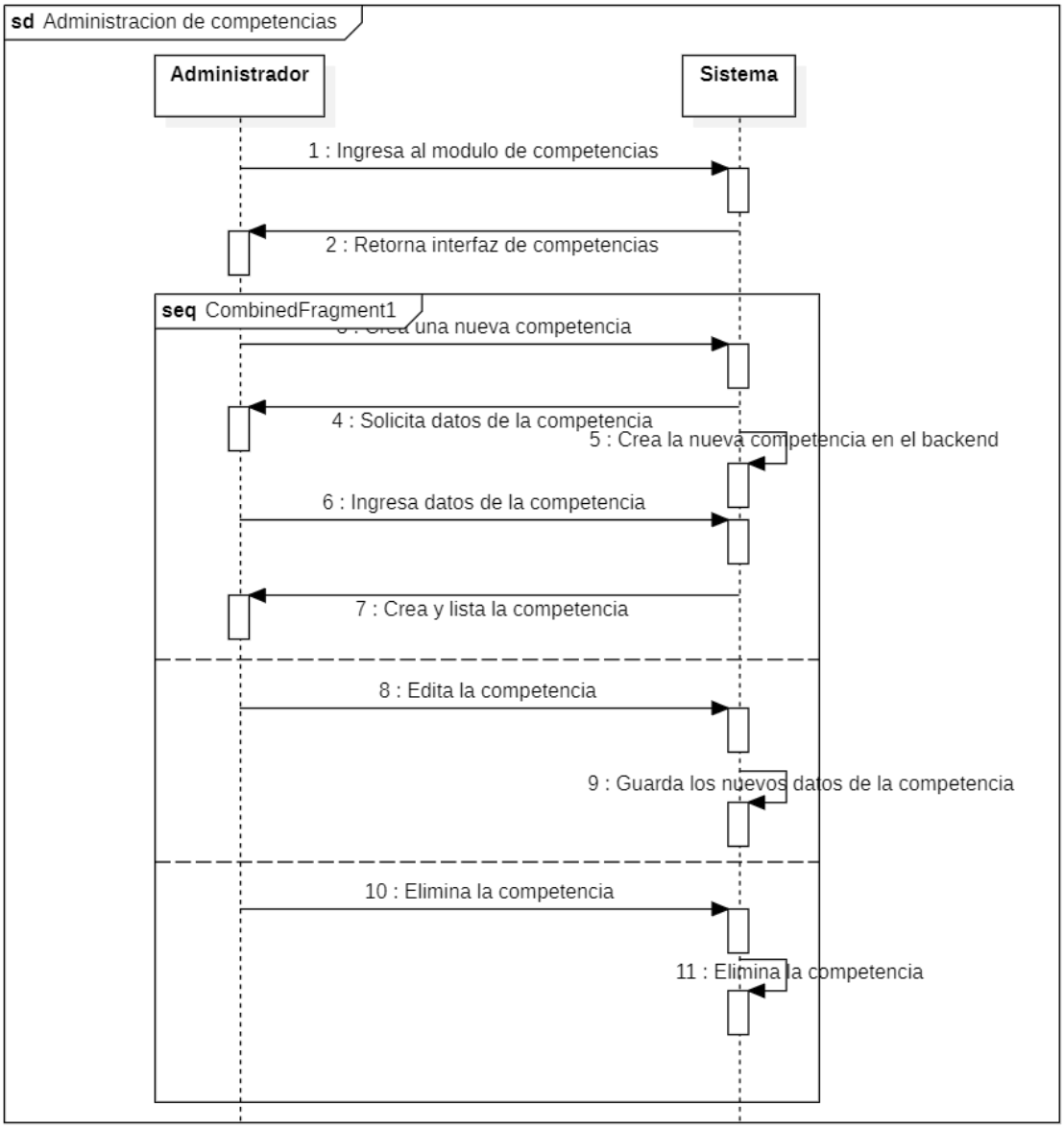


Figura 3.14

Diagrama de secuencia: Administración de Competencias

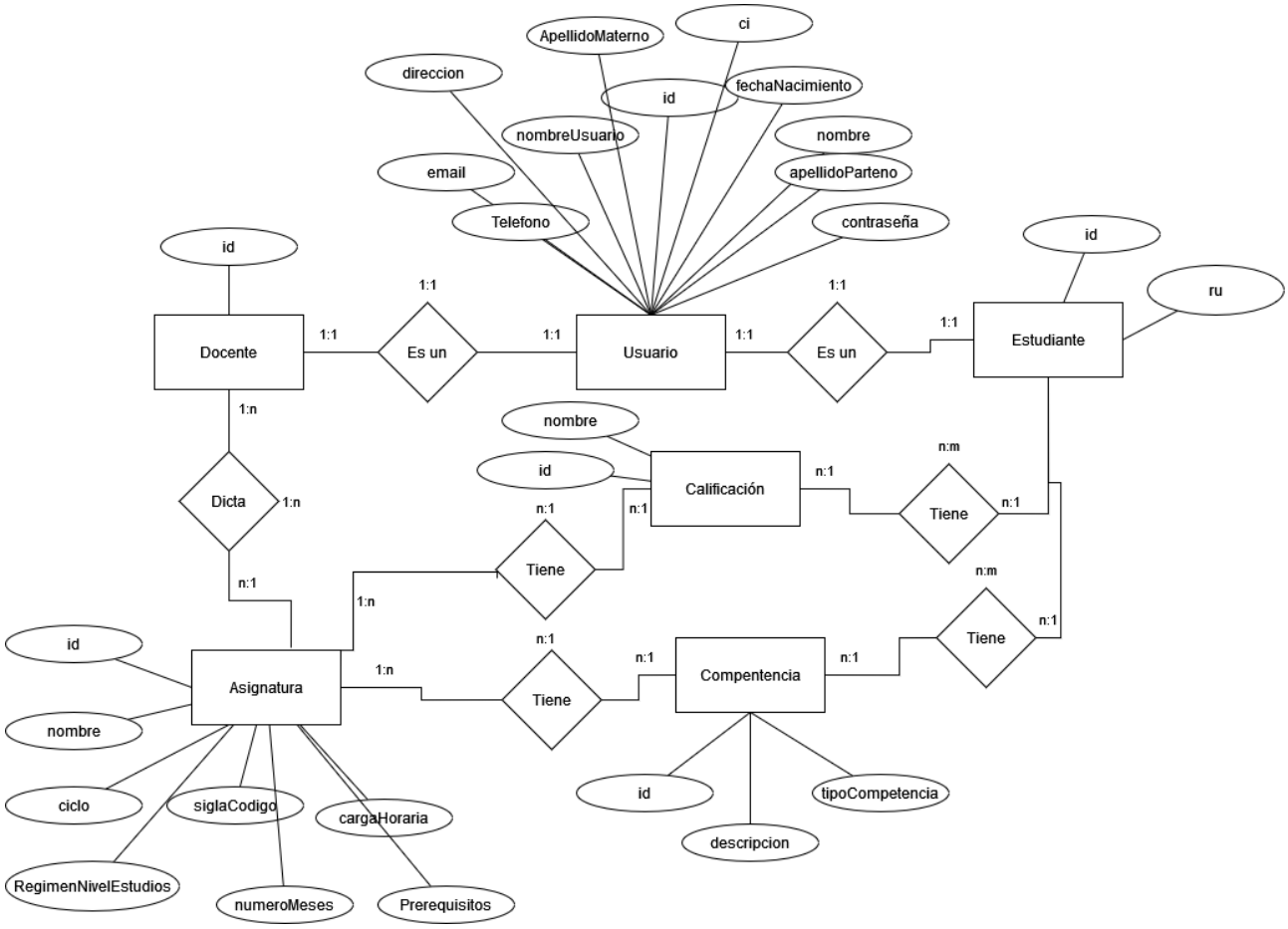


3.2.4 Diagrama Entidad Relación

En siguiente diagrama se realiza el modelado de datos del sistema mediante entidades las cuales son: Docente, Estudiante, Usuario, Calificación, Asignatura y Competencia.

Figura 3.15

Diagrama Entidad Relación General

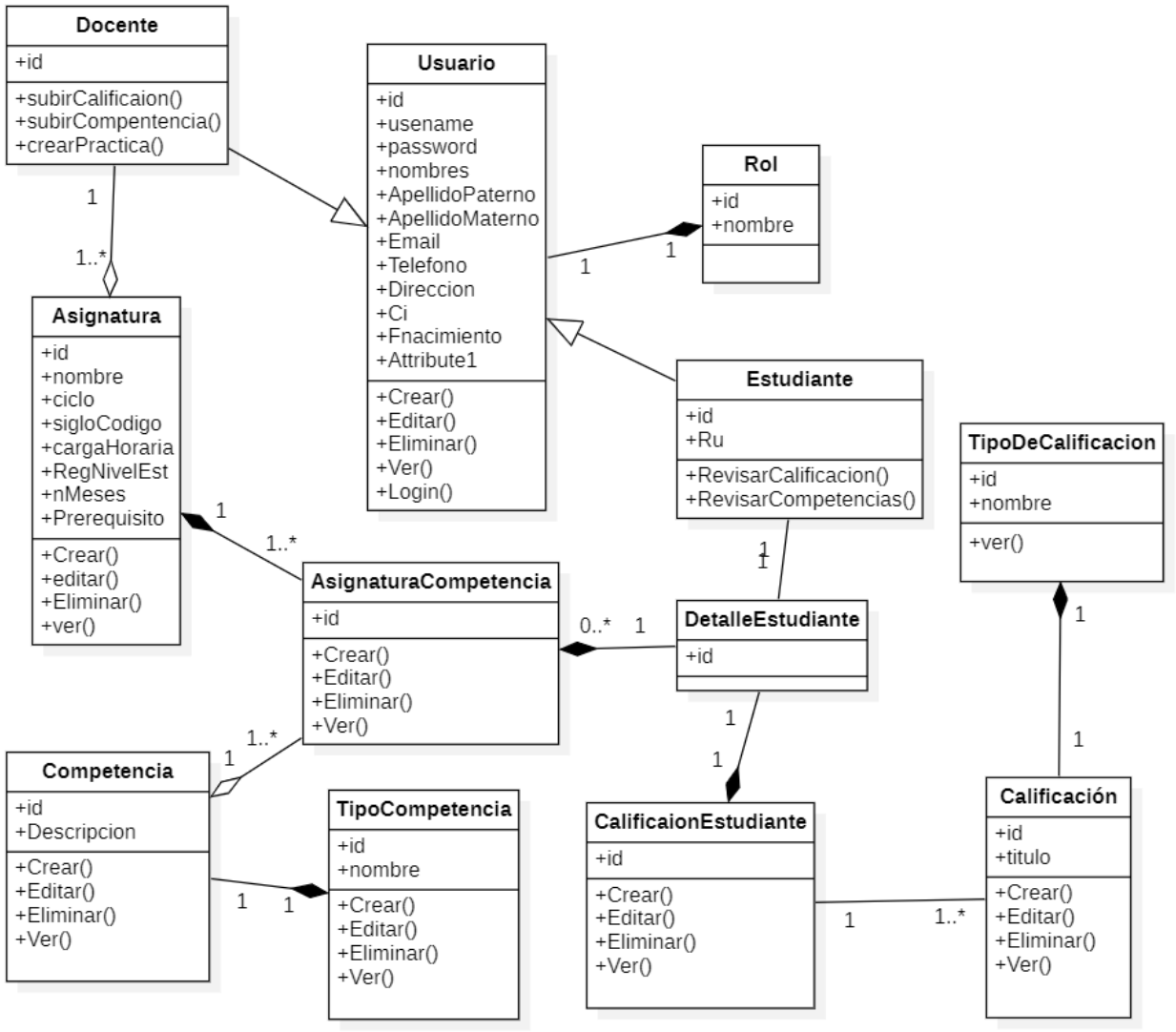


Nota. Muestra las entidades que se desarrollaran y así también se definen sus propiedades

3.2.5 Diagrama de Clases

Figura 3.16

Diagrama de clases del sistema

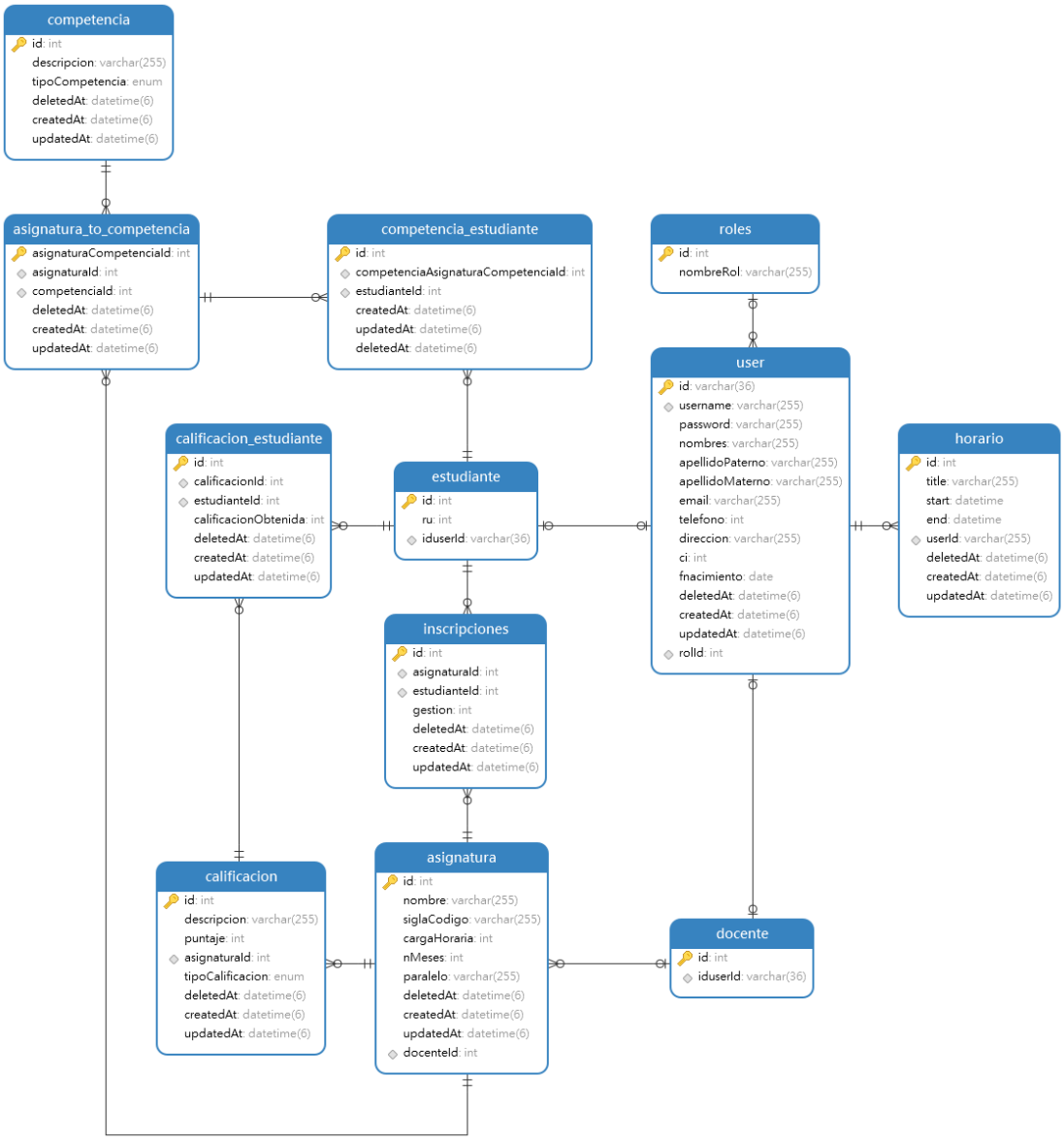


Nota. Se muestra la estructura estática que describe la estructura de un sistema mostrando las clases del sistema, sus atributos y métodos

3.2.6 Diagrama físico de la base de datos

Figura 3.17

Diagrama físico de la base de datos



Nota. Diagrama físico de la base de datos con sus llaves primarias y llaves foráneas

3.2.7 Diseño navegacional de pantallas

En el desarrollo del diseño navegacional se busca que el usuario pueda localizar la información que necesita de una manera fácilmente reconocible y accesible de manera que se tenga una experiencia adecuada y de esta manera se pueda llegar a la satisfacción del usuario a continuación en las figuras 3.5 a 3.14 se muestra el diseño inicial de la interfaz la cual contempla los módulos requeridos para los diferentes roles: administrador, docente y estudiante

Figura 3.18

Login de inicio de sesión

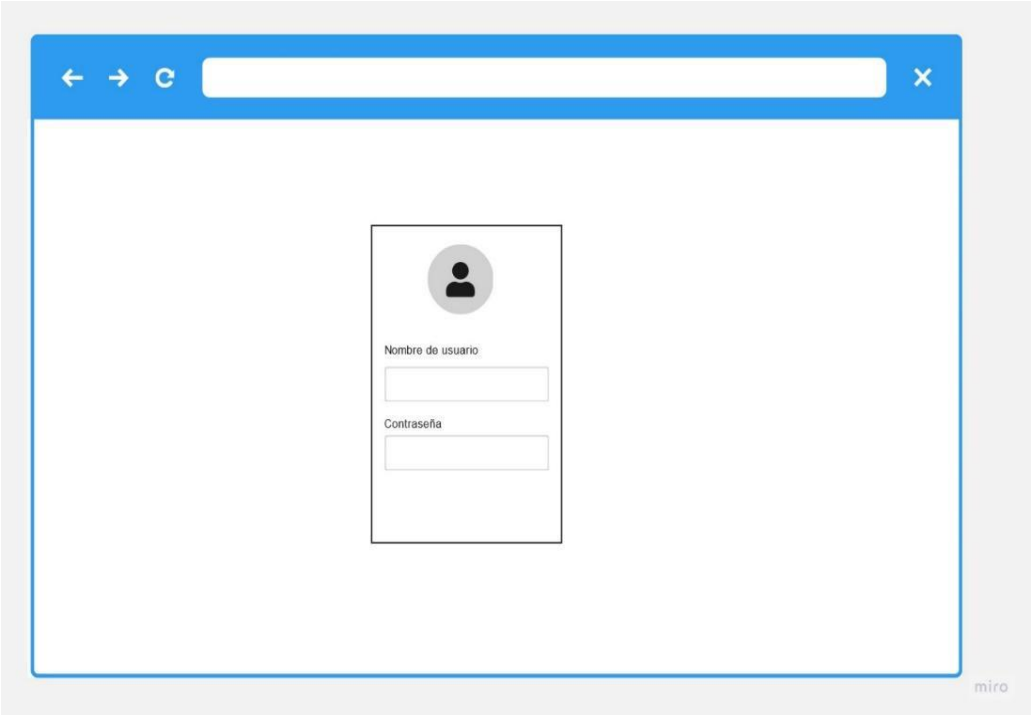


Figura 3.19

Pantalla de inicio con los módulos accesibles por el rol

Figura 3.20

Módulo de estudiantes

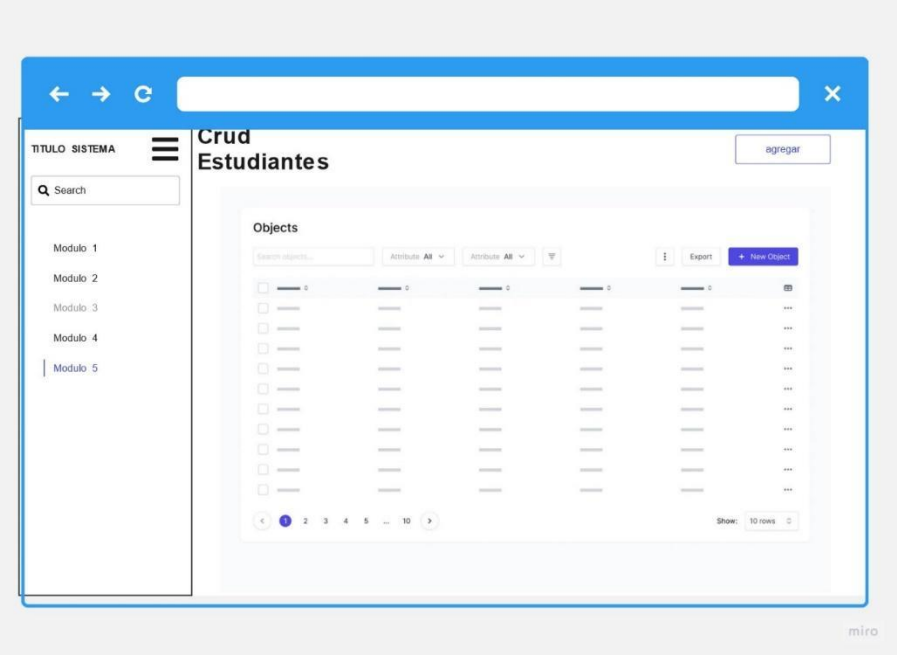


Figura 3.21

Módulo de materias

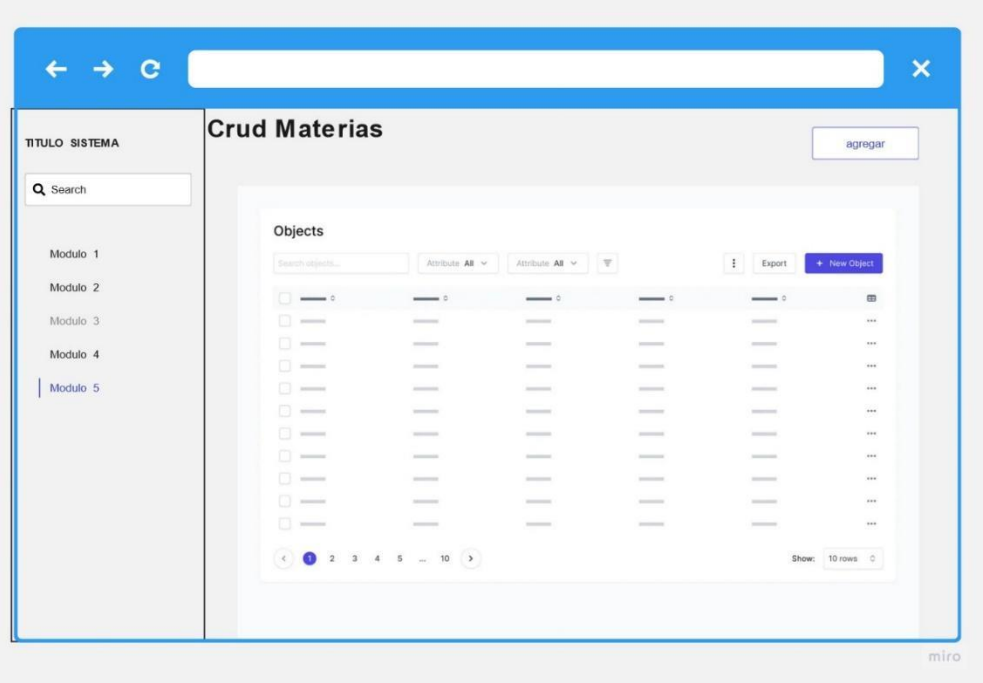


Figura 3.22

Módulo de docentes

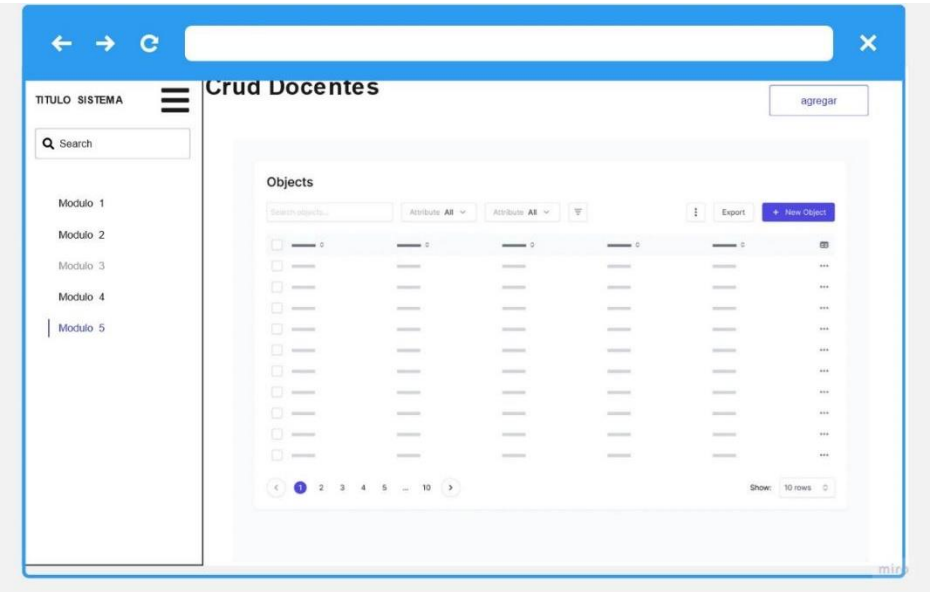


Figura 3.23

Módulo de competencias

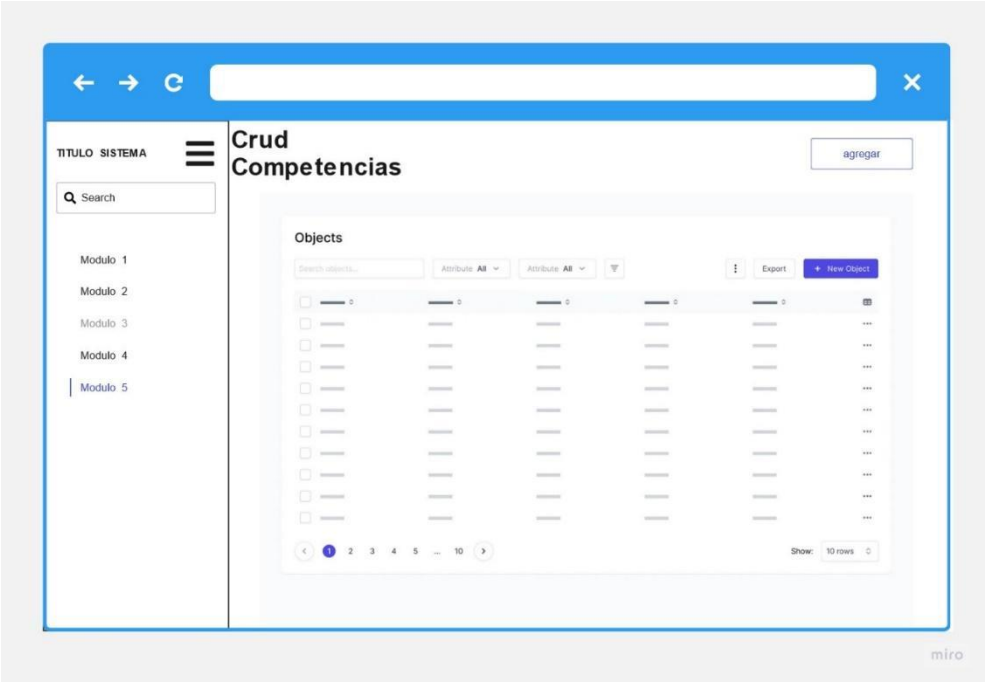


Figura 3.24

Módulo de inscripciones

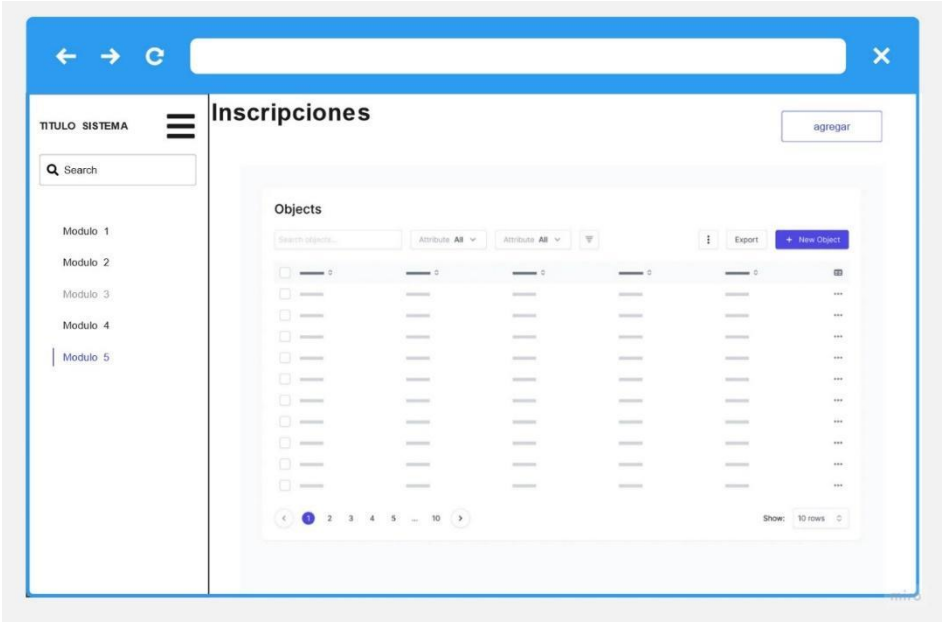


Figura 3.25

Login e Inicio en diseño responsivo

Figura 3.26

Materias de docente y Administración de materia

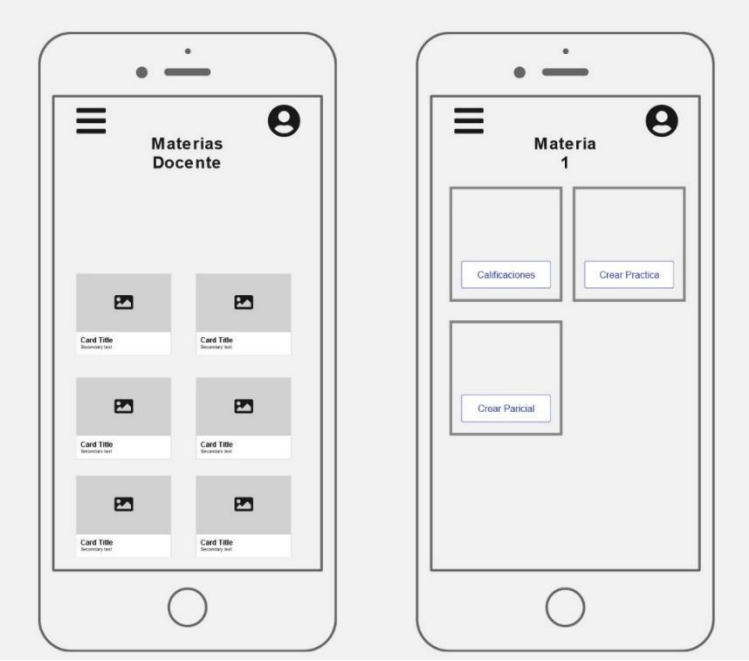


Figura 3.27

Módulo de detalle de estudiante

3.2.8 Diagrama Navegacional

Figura 3.28

Diagrama navegacional de Docentes

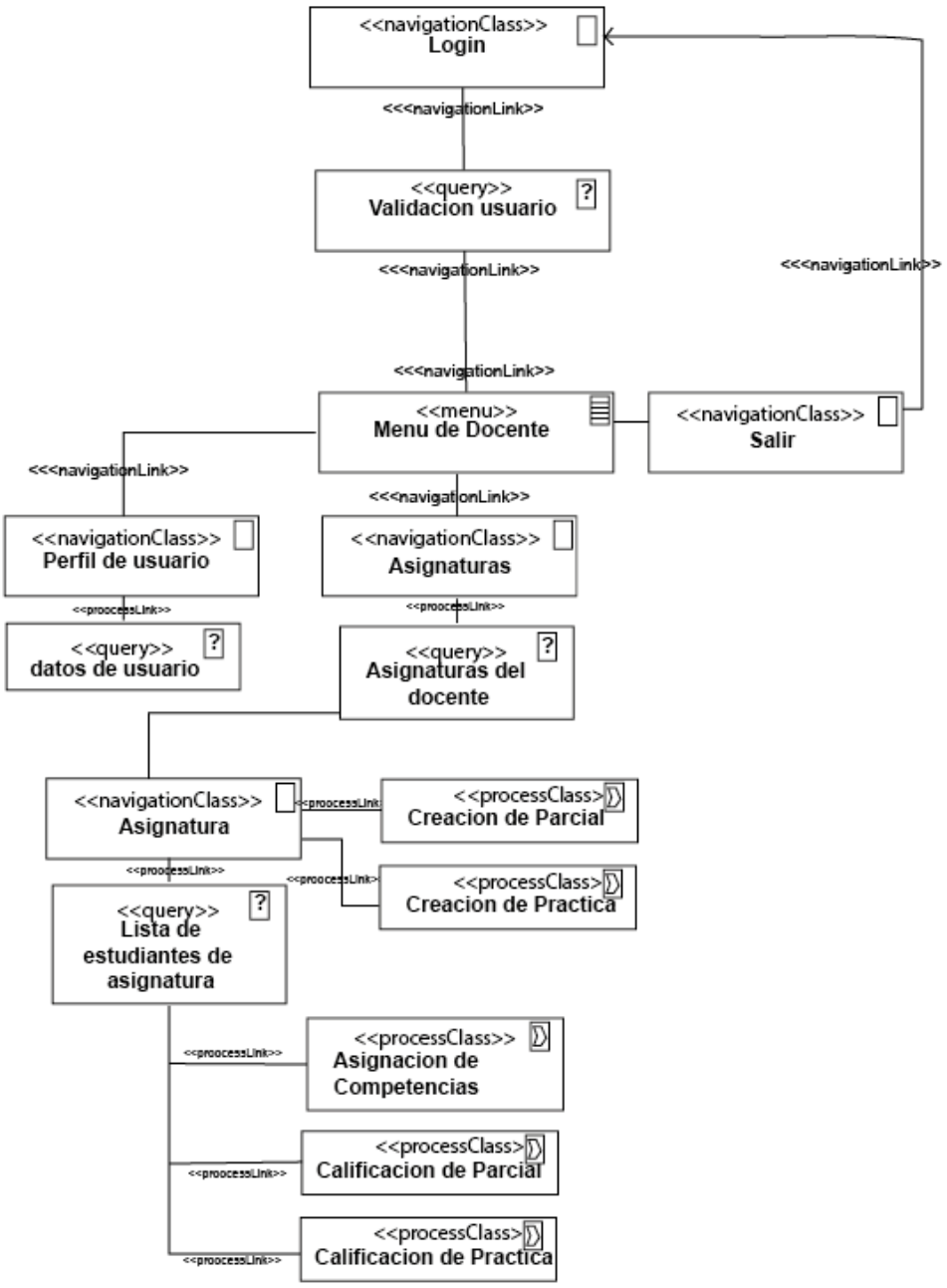


Figura 3.29

Diagrama navegacional estudiantes

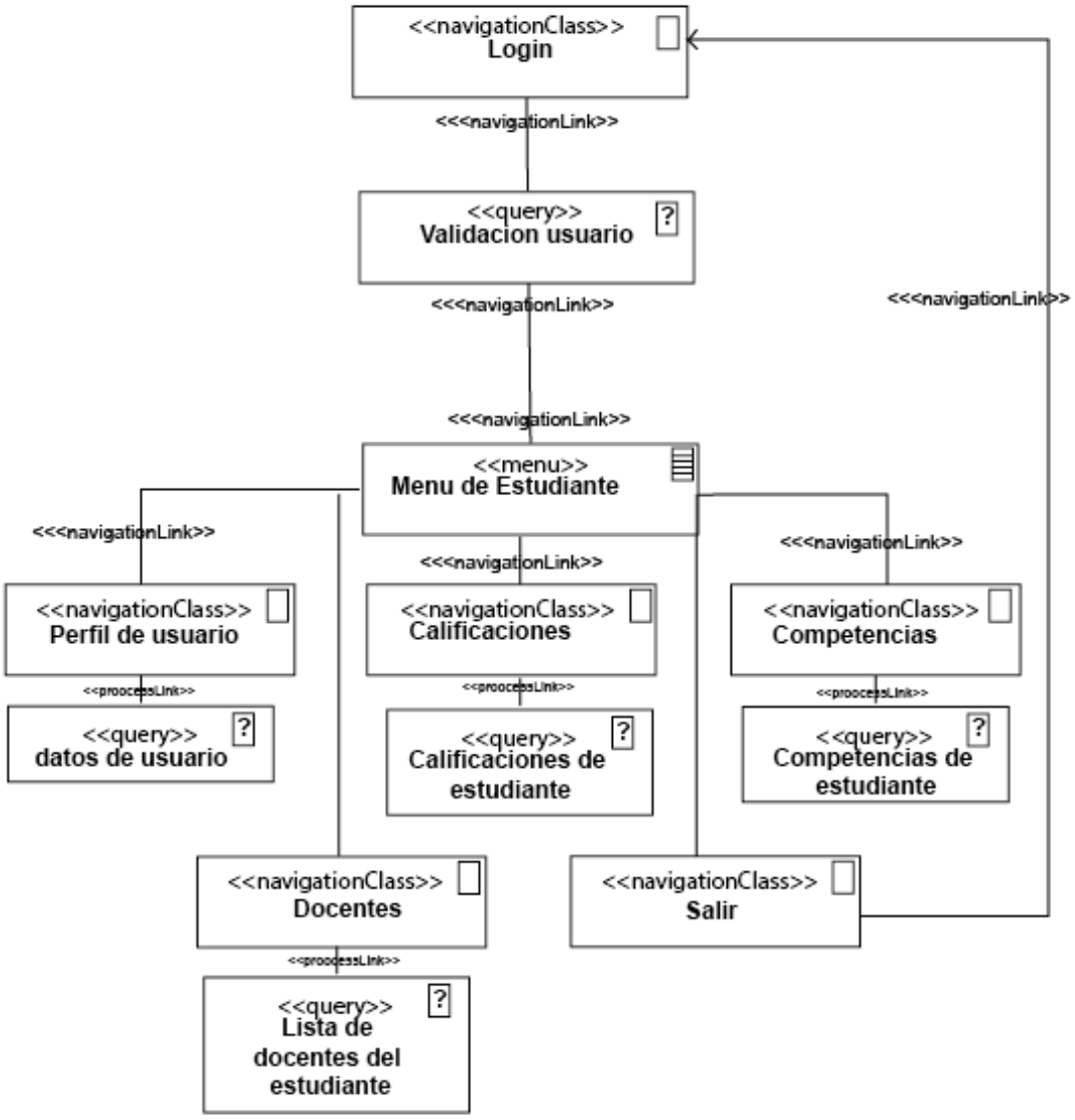
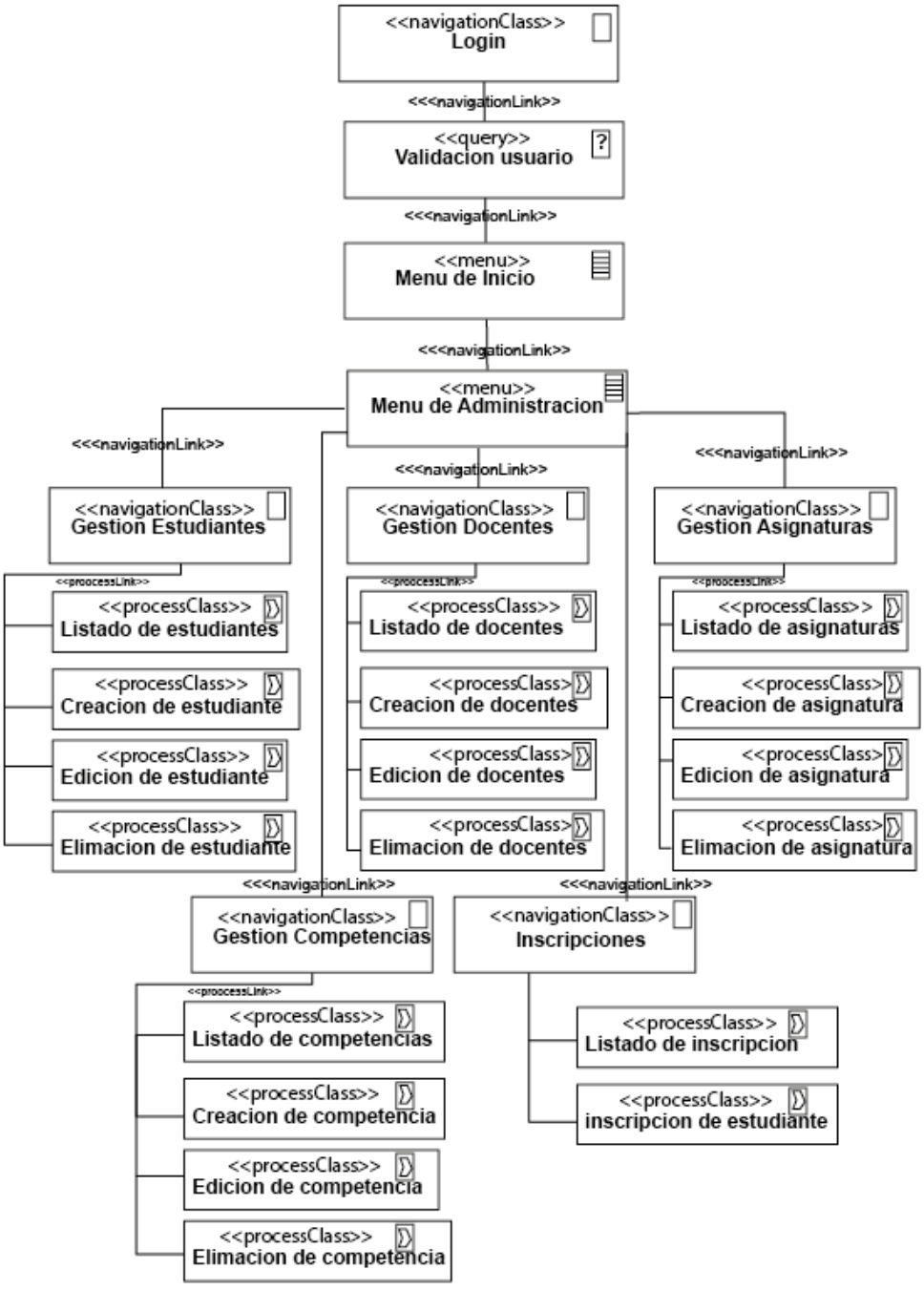


Figura 3.30

Diagrama navegacional administrado



3.3 Fase de Codificación

En la fase de codificación de la metodología XP, el equipo de desarrollo se encarga de implementar las historias de usuario que fueron priorizadas en la fase de planificación.

3.3.1 Backend

En el desarrollo del backend se utilizó como framework Nestjs para el desarrollo de los casos de uso se dividió en los siguientes módulos.

3.3.1.1 Modulo app

El módulo es la raíz del proyecto donde se instancian las librerías y se llaman a los demás modelos para su ejecución, a continuación, se describe fracciones del código del módulo app.

```
@Module({
  imports: [
    ConfigModule.forRoot({ isGlobal: true }),
    TypeOrmModule.forRoot({
      type: 'mysql',
      host: process.env.DATABASE_HOST,
      port: parseInt(process.env.DATABASE_PORT),
      username: process.env.DATABASE_USER,
      password: process.env.DATABASE_PASSWORD,
      database: process.env.DATABASE_NAME,
      entities: [
        User,
        Docente,
        Asignatura,
        Roles,
        Estudiante,
        Competencia,
        AsignaturaToCompetencia,
        Inscripciones,
        CompetenciaEstudiante,
```

```
        Calificacion,
        CalificacionEstudiante,
        Horario
      ],
      synchronize: true,
    }),
    UserModule,
    PostModule,
    CompetenciaModule,
    AsignaturaModule,
    EstudianteModule,
    DocenteModule,
    CalificacionModule,
    HorariosModule,
  ],
  controllers: [AppController],
  providers: [AppService],
})
```

3.3.1.2 Modulo usuario

Se muestra en código para la creación de usuarios

```
async create(createUserDto: CreateUserDto) {
    const { fnacimiento } = createUserDto;
    const { nombres } = createUserDto;
    const primernombre = nombres.split(' ');
    const { ci } = createUserDto;
    const { rol } = createUserDto;
    const passCryps = await hash(ci.toString(), 10);
     createUserDto = { ...createUserDto, username: primernombre[0] + '_' + ci
,password:passCryps};
    const userFound = await this.userRepository.findOne({
      where: {
        username: primernombre[0] + '_' + ci,
      },
```

```
    });
    if (userFound) {
      return new HttpException('Usuario ya existe', HttpStatus.CONFLICT);
    }

    const newUser = await this.userRepository.create(createUserDto);

    const savedUser = await this.userRepository.save(newUser);
    const datosEstudiante = {
      ru: createUserDto.ru,
      iduser: savedUser.id,
    };
    const id = { iduser: savedUser.id };
    if (JSON.stringify(savedUser.rol) == '1') {
      this.docenteRepository.save(id);
    }
    if (JSON.stringify(savedUser.rol) == '3') {
      this.estudianteRepository.save(datosEstudiante);
    }
    return savedUser;
  }
```

A continuación se muestra el controlador para el manejo de las solicitudes de usuario.

```
@ApiTags('user')
@ApiBearerAuth()
@Controller('user')
export class UserController {
  constructor(private readonly userService: UserService) {}
  @UseGuards(JwtAuthGuard)
  @Get('/profile')
  findallprofile() {
    return this.userService.getProfiles();
  }
```

```
@Post('/login')
login(@Body() loginUserDto: LoginUserDto) {
  return this.userService.login(loginUserDto);
}

// @UseGuards(JwtAuthGuard)
@Post()
create(@Body() createUserDto: CreateUserDto) {
  return this.userService.create(createUserDto);
}

@UseGuards(JwtAuthGuard)
@Get()
findAll(): Promise<User[]> {
  return this.userService.findAll();
}

@UseGuards(JwtAuthGuard)
@Get(':id')
findOne(@Param('id') id: string) {
  return this.userService.findOne(id);
}

@UseGuards(JwtAuthGuard)
@Patch(':id')
update(@Param('id') id: string, @Body() updateUserDto: UpdateUserDto) {
  return this.userService.update(id, updateUserDto);
}

@UseGuards(JwtAuthGuard)
@Delete(':id')
remove(@Param('id') id: string) {
  return this.userService.remove(id);
}
```

```
  @UseGuards(JwtAuthGuard)
  @Post(':id/profile')
  createProfile(
    @Param('id') id: string,
    @Body() createProfileDto: CreateProfileDto,
  ) {
    return this.userService.createProfile(id, createProfileDto);
  }

  @UseGuards(JwtAuthGuard)
  @Post('/changepass')
  change_pass(@Body() changePasswordDto: ChangePasswordDto) {
    return this.userService.changepassword(changePasswordDto);
  }
}
```

3.3.2 Frontend

3.3.2.1 Módulo de inicio de Sesión

```
<div>
  <p-messages
    [(value)]="messages1"
    [enableService]="false"
    [closable]="false"
  ></p-messages>
</div>

<div class="flex align-items-center align-content-center justify-content-center
m-3 max-w-screen scalein animation-duration-300">
  <div
    class="flex flex-column surface-card shadow-2 border-round p-3 m-3 w-full
md:w-5"
```

```
  >
    <div class=" flex flex-column zoom text-center mb-3">
      <!-- <img
        src="../../assets/images/Happy student-rafiki.png"
        alt="Image"
        height="200"
        class=""
      /> -->
      <object
        class="fondo2"
        data="../../assets/images/happy-student-animate.svg"
        type=""
        height="200"
      ></object>
      <div class="text-900 text-3xl font-medium mb-1">Bienvenido</div>
       <!-- <span class="text-600 font-medium line-height-3"> ¿No tienes una
cuenta?</span> -->
        <!-- <a class="font-medium no-underline ml-2 text-blue-500 cursor-
pointer">Creala hoy!</a> -->
            <div   class="mb-1"><h1   class="text-900   text-3xl   font-bold
">SISEIN</h1></div>
      <div class="text-900 text-l font-normal mb-1">
        SISTEMA DE SEGUIMIENTO INDIVIDUAL
      </div>

      <span class="text-600 font-medium line-height-3"
        >No tienes una cuenta?</span
      >
      <a
        href="https://forms.gle/URVZiPVBgjTsK5Ct7"
        target="_blank"
        class="font-medium no-underline ml-2 text-blue-500 cursor-pointer"
        >Solicita una Ahora!</a
      >
    </div>
```

```
    <div>
      <form (ngSubmit)="onSubmit()" [formGroup]="loginForm" class="p-2">
        <label for="email1" class="block text-900 font-medium mb-2"
          >Nombre de Usuario</label
        >
        <input
          id="username"
          type="text"
          placeholder="Nombre de usuario"
          pInputText
          class="w-full mb-3"
          formControlName="username"
          required
                 [ngClass]="{'ng-dirty':loginForm.get('username')?.touched   &&
loginForm.get('username')?.errors?.['required']}"
        />
        <div
                                *ngIf="loginForm.get('username')?.touched     &&
loginForm.get('username')?.errors?.['required']"
        >
          <p-messages
            [(value)]="messages"
            [closable]="false"
            [showTransitionOptions]="'50ms'"
            [hideTransitionOptions]="'500ms'"
          ></p-messages>
        </div>

        <label for="password1" class="block text-900 font-medium mb-2"
          >Contraseña</label
        >
        <input
          id="password"
          type="password"
```

```
          placeholder="Contraseña"
          pInputText
          class="w-full mb-3"
          formControlName="password"
          required
                 [ngClass]="{'ng-dirty':loginForm.get('password')?.touched   &&
loginForm.get('password')?.errors?.['required']}"
        />
        <div
                                *ngIf="loginForm.get('password')?.touched     &&
loginForm.get('password')?.errors?.['required']"
        >
          <p-messages
            [(value)]="messages"
            [closable]="false"
            [showTransitionOptions]="'50ms'"
            [hideTransitionOptions]="'500ms'"
          ></p-messages>
        </div>
        <div class="flex align-items-center justify-content-between mb-6">
          <div class="flex align-items-center">
            <p-checkbox
              id="rememberme1"
              [binary]="true"
              styleClass="mr-2"
              formControlName="remember"
            ></p-checkbox>
            <label for="rememberme1" class="text-900">Recuerdame</label>
          </div>
          <a
           class="font-medium no-underline ml-2 text-blue-500 text-right cursor-
pointer"
            >¿Olvidaste tu contraseña?</a
          >
        </div>
```

```
        <button
          pButton
          pRipple
          type="submit"
          label="Ingresar"
          icon="pi pi-user"
          class="w-full"
        ></button>
      </form>
    </div>
  </div>
</div>
```

3.3.3 Interfaz de usuario

3.3.3.1 Primera Iteración

Figura 3.31

Vista Interfaz de login

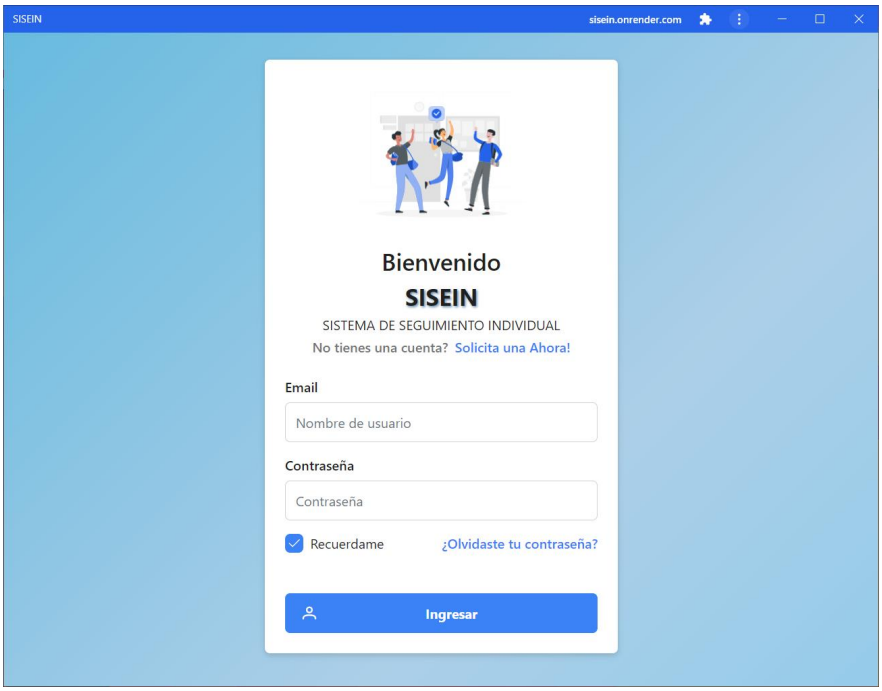


Figura 3.32

Vista Panel de administración

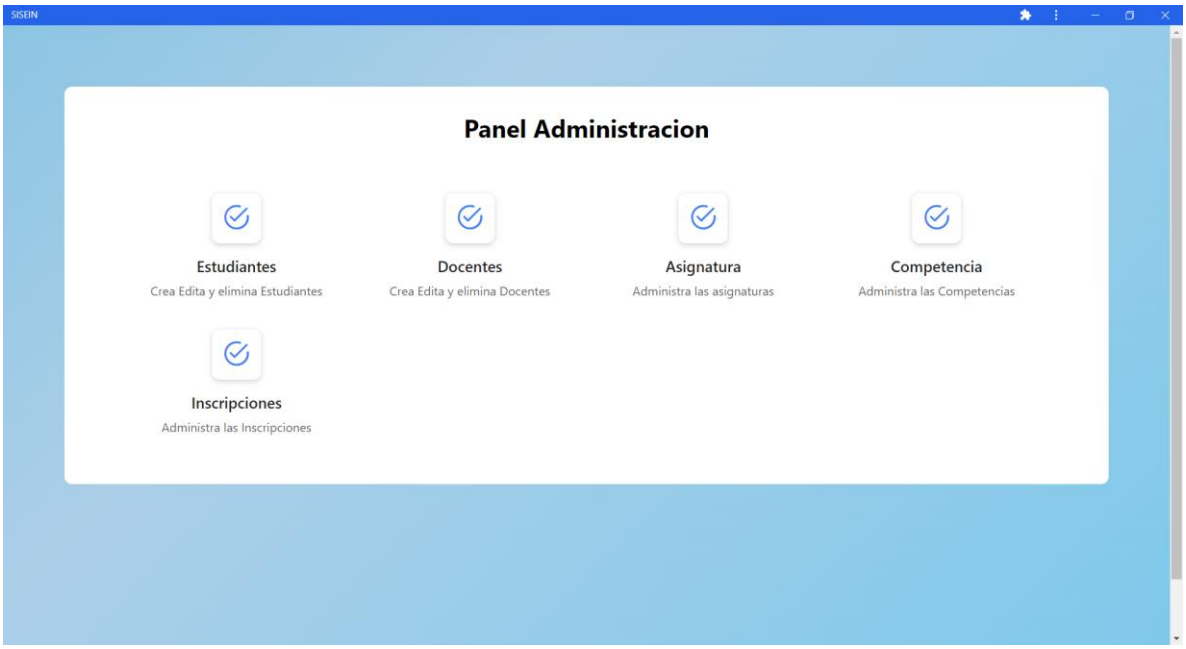
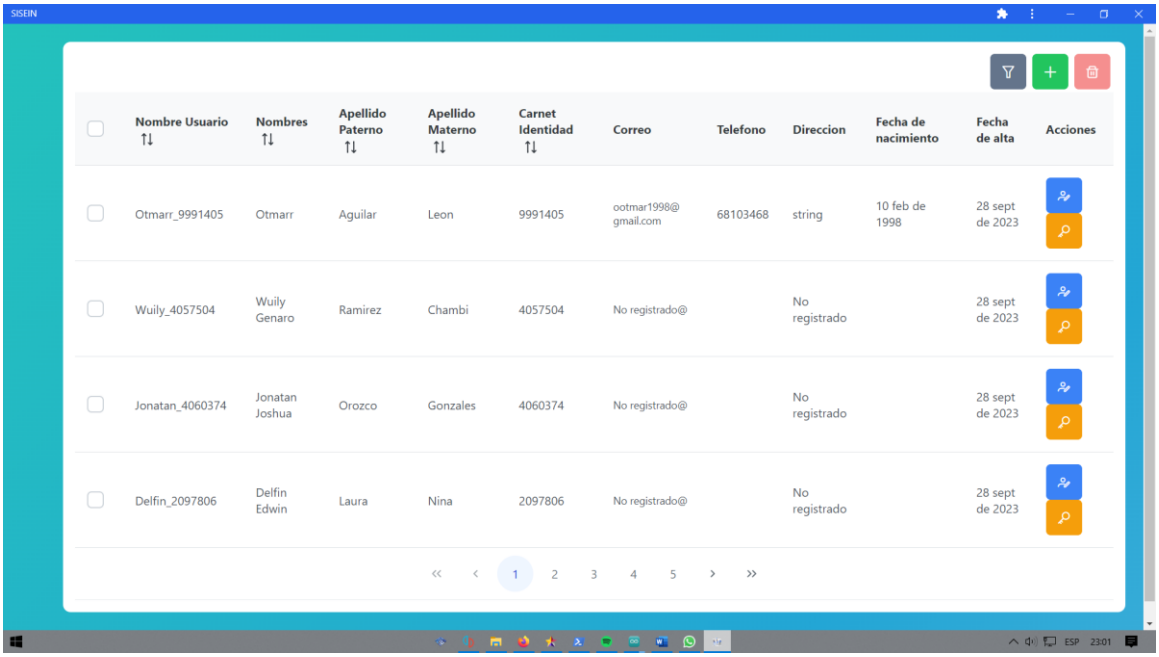


Figura 3.33

Vista Panel de administración de estudiantes



3.3.3.2 Segunda Iteración

Figura 3.34

Vista Panel de administración de docentes

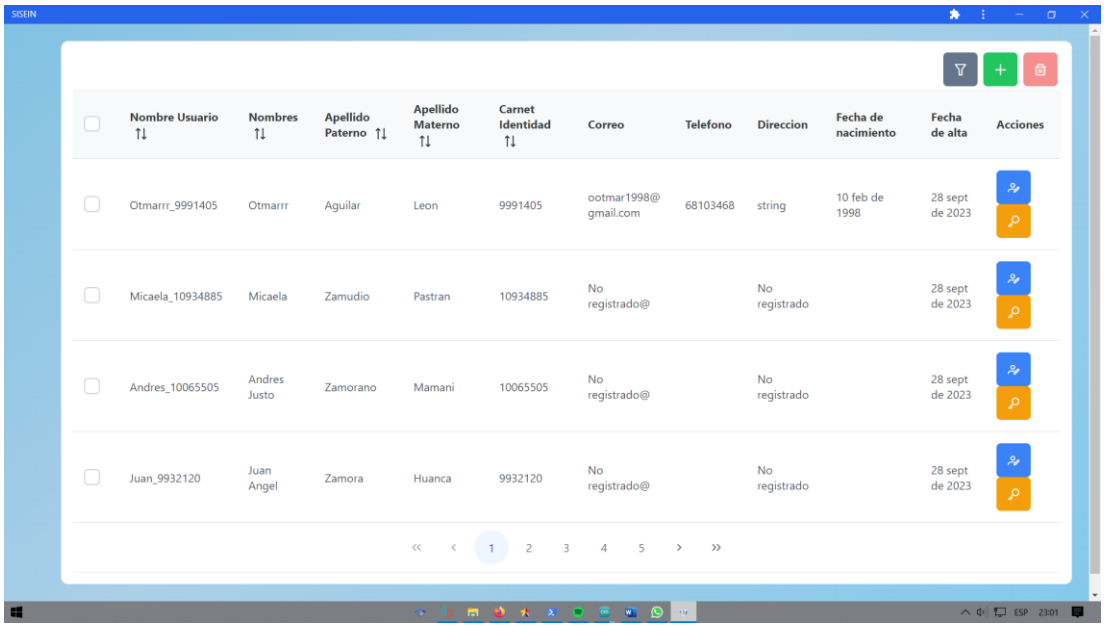


Figura 3.35

Vista Panel de administración de asignaturas

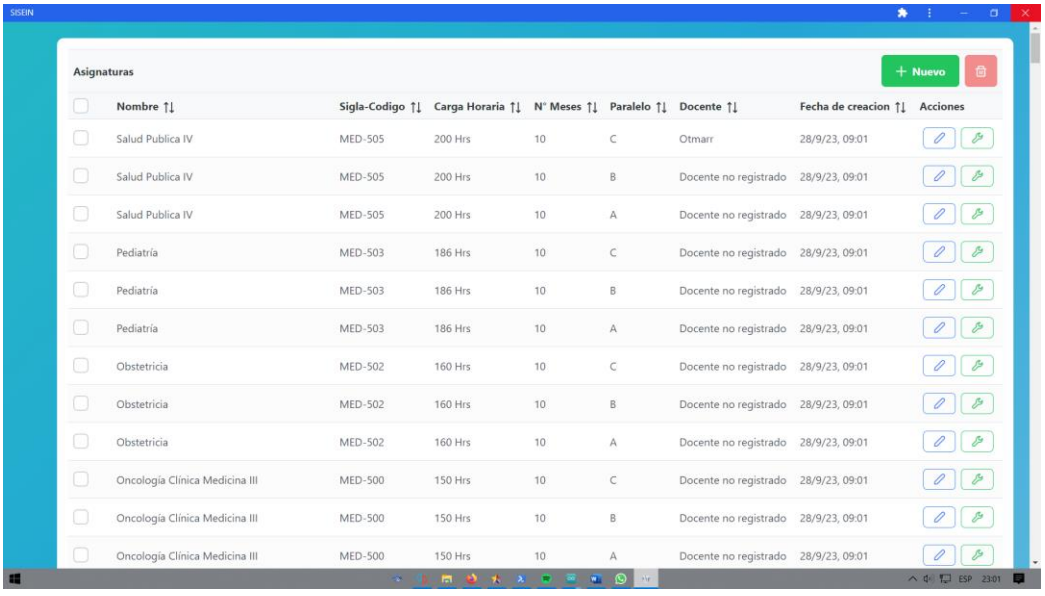
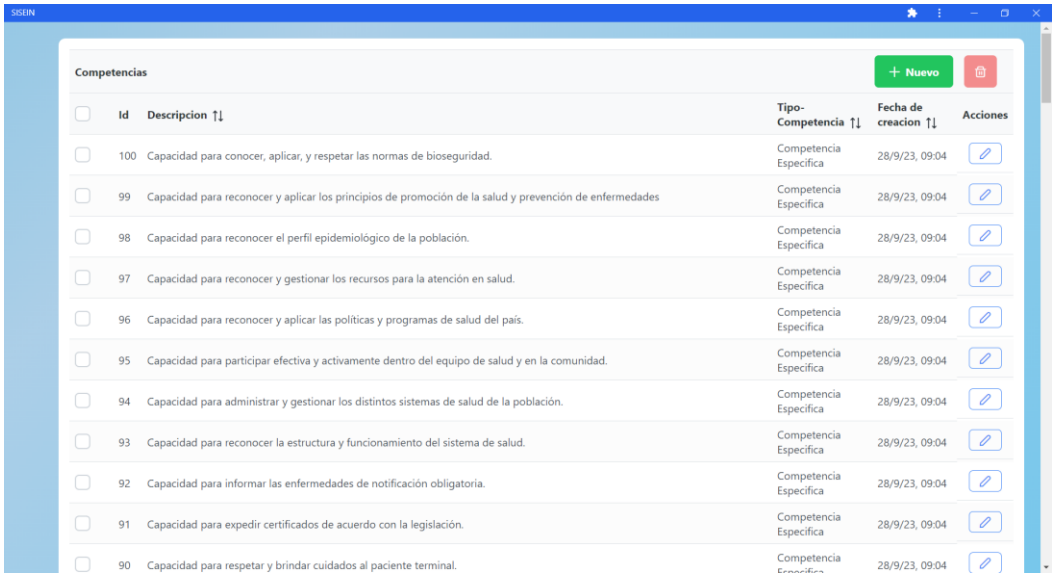


Figura 3.36

Vista Panel de administración de competencias



3.3.3.3 Tercera iteración

Figura 3.37

Listado de estudiantes

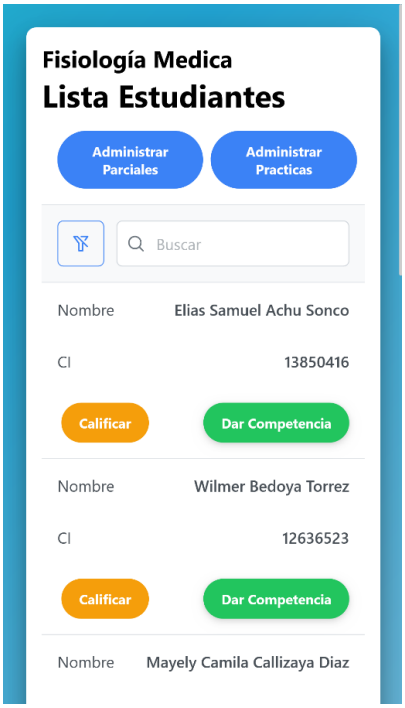


Figura 3.38

Vista Panel de administración de Parciales

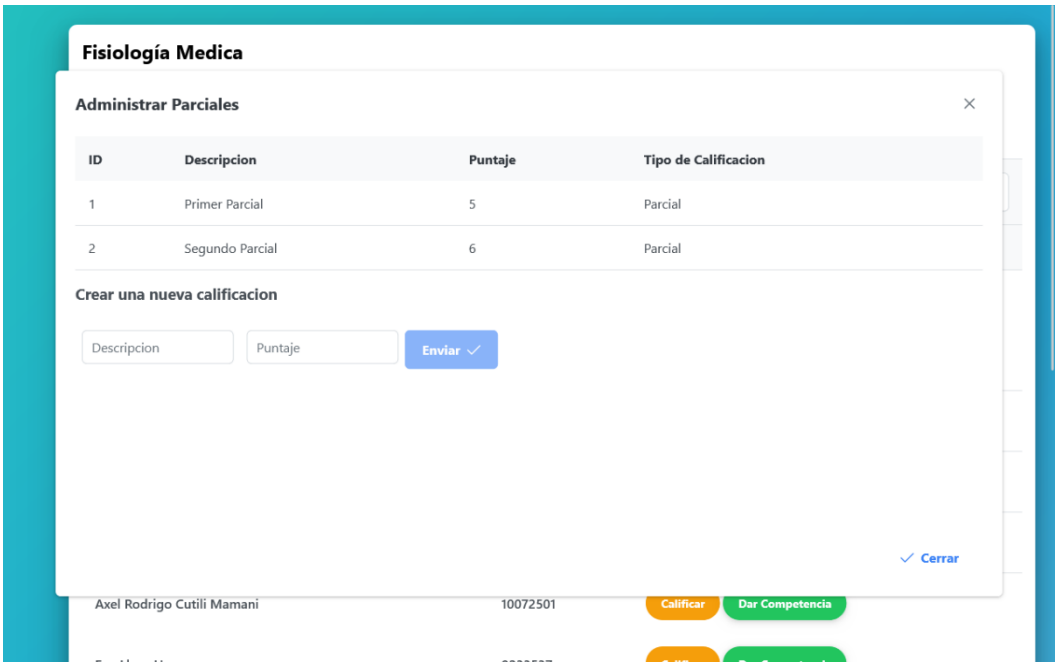


Figura 3.39

Vista Panel de administración de practicas

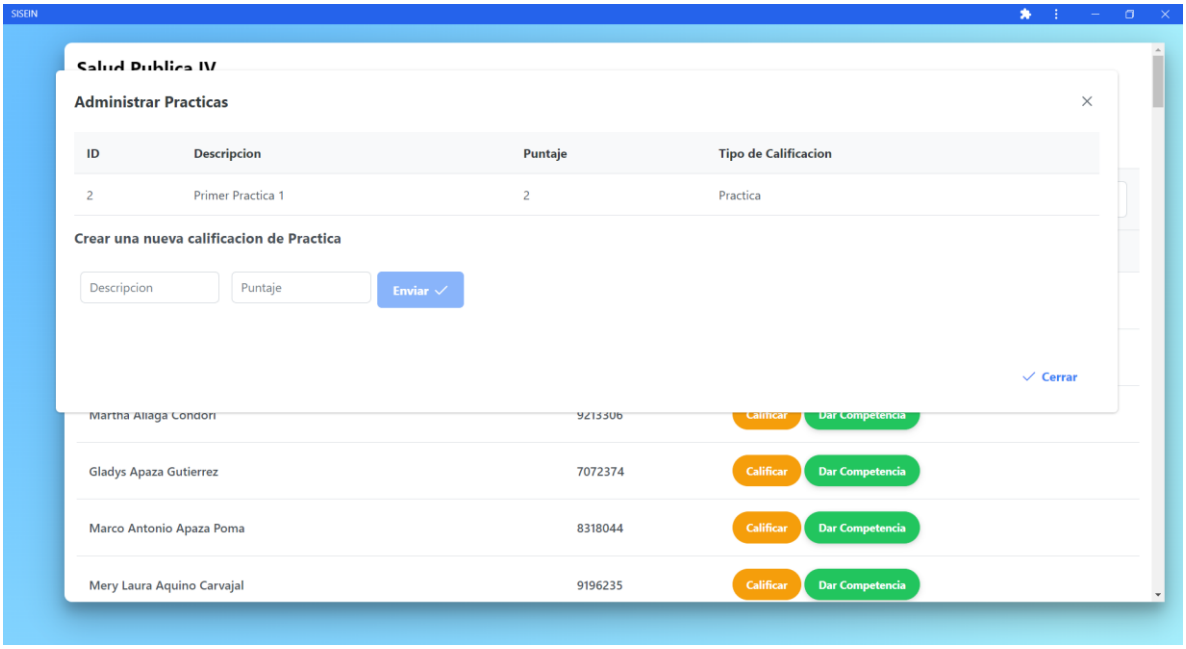


Figura 3.40

Panel de asignaturas de docente

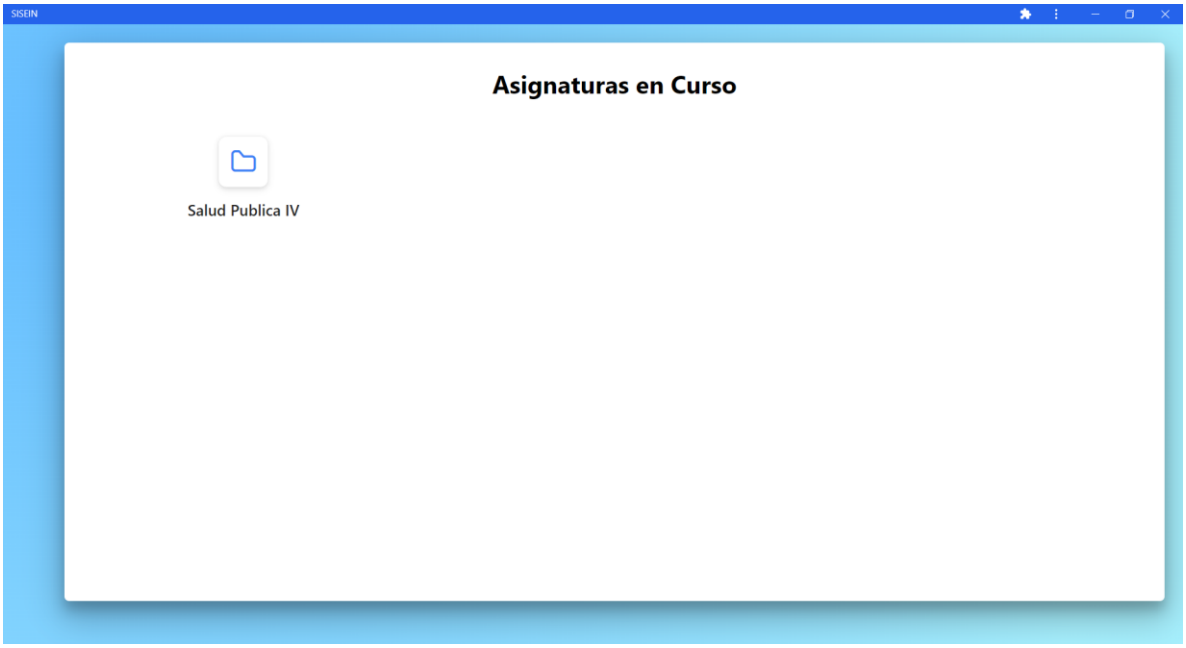


Figura 3.41

Calificar estudiante

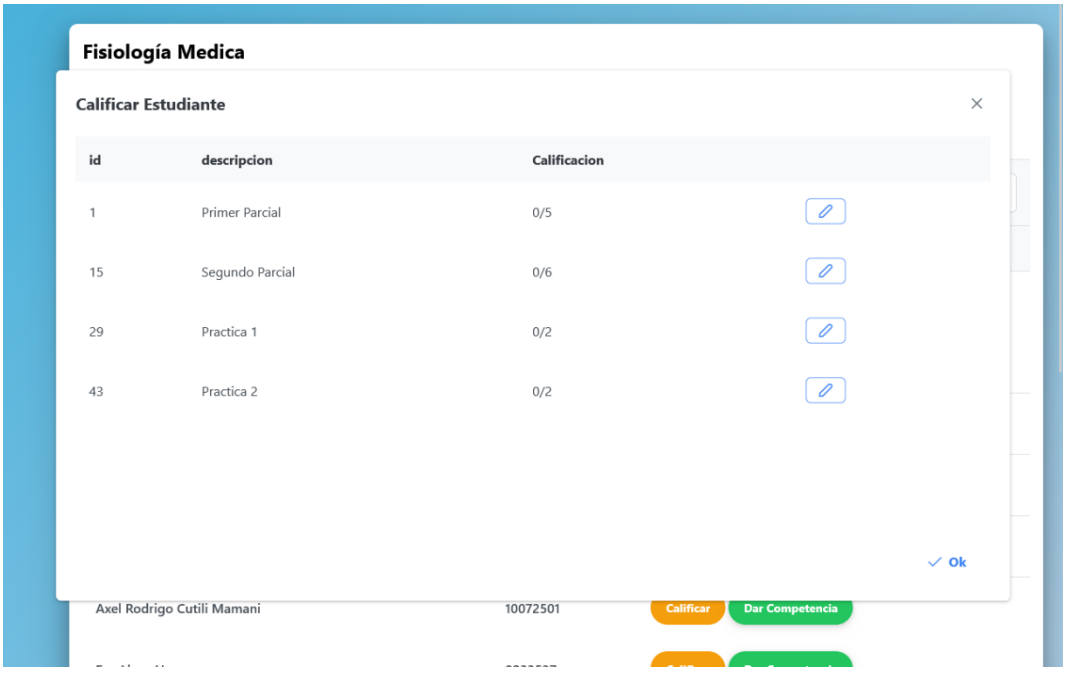


Figura 3.42

Dialogo de introducción de calificación

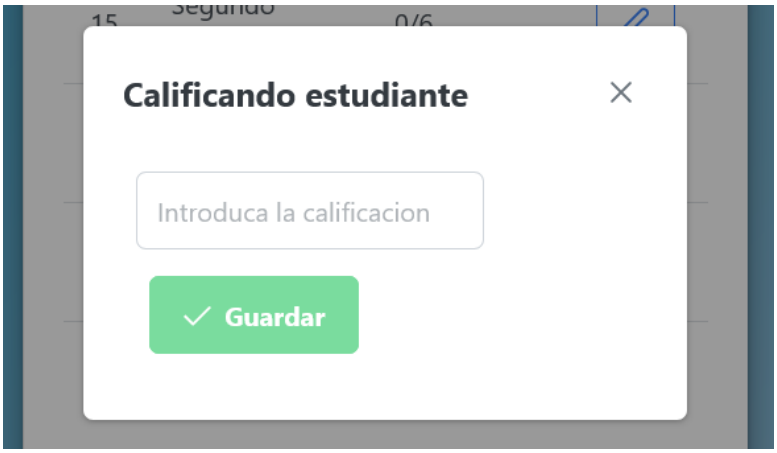
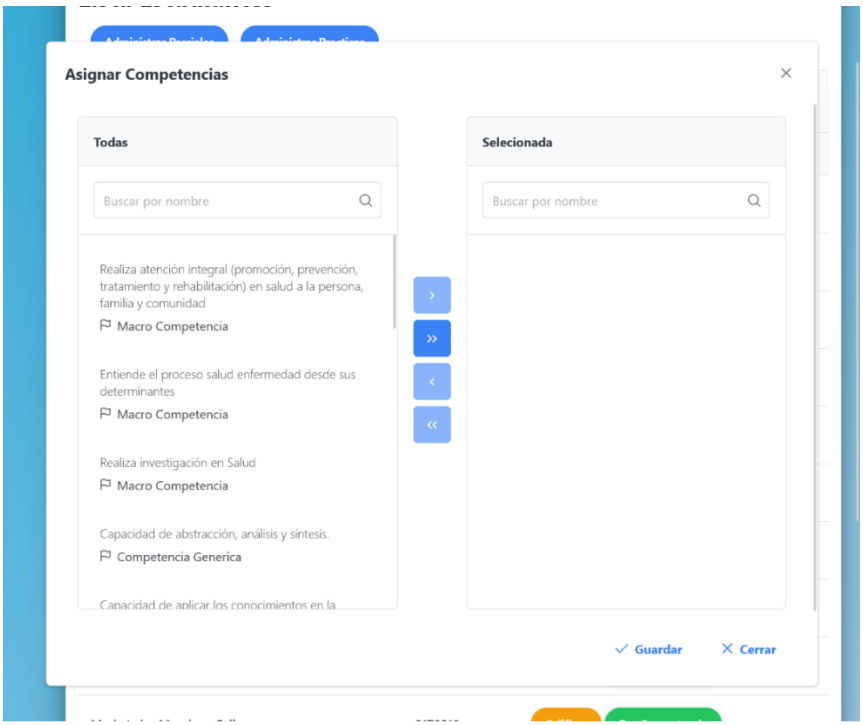


Figura 3.43

Asignación de competencia a estudiante



3.3.3.4 Cuarta Iteración

Figura 3.44

Vista Inicio del sistema estudiante

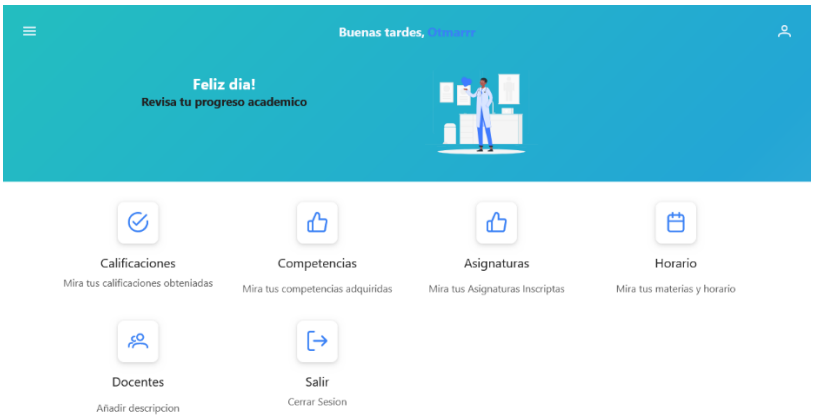


Figura 3.45

Vista Inicio del sistema estudiante responsive

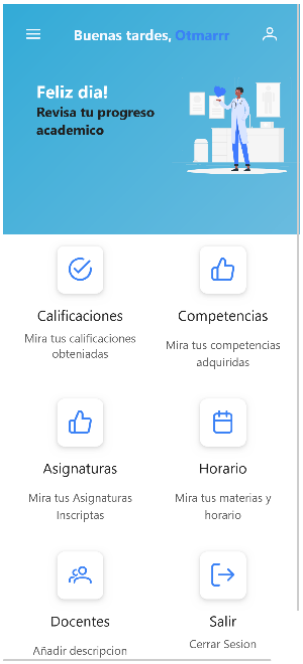


Figura 3.46

Perfil usuario

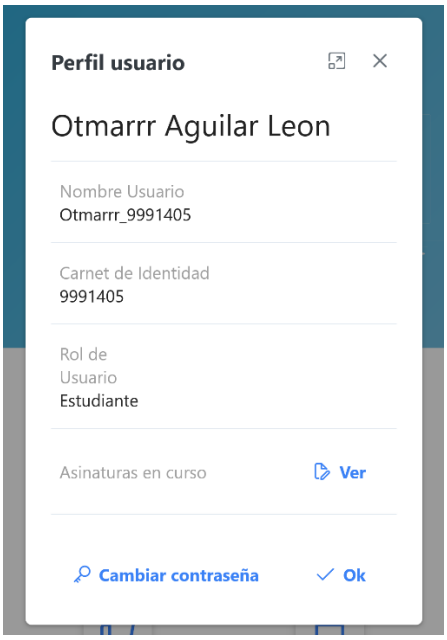


Figura 3.47

Calificaciones de estudiante (vista responsive)

Figura 3.48

Competencia de estudiante (vista responsive)

Figura 3.49

Asignaturas de estudiante (vista responsive)

Figura 3.50

Módulo de inscripciones

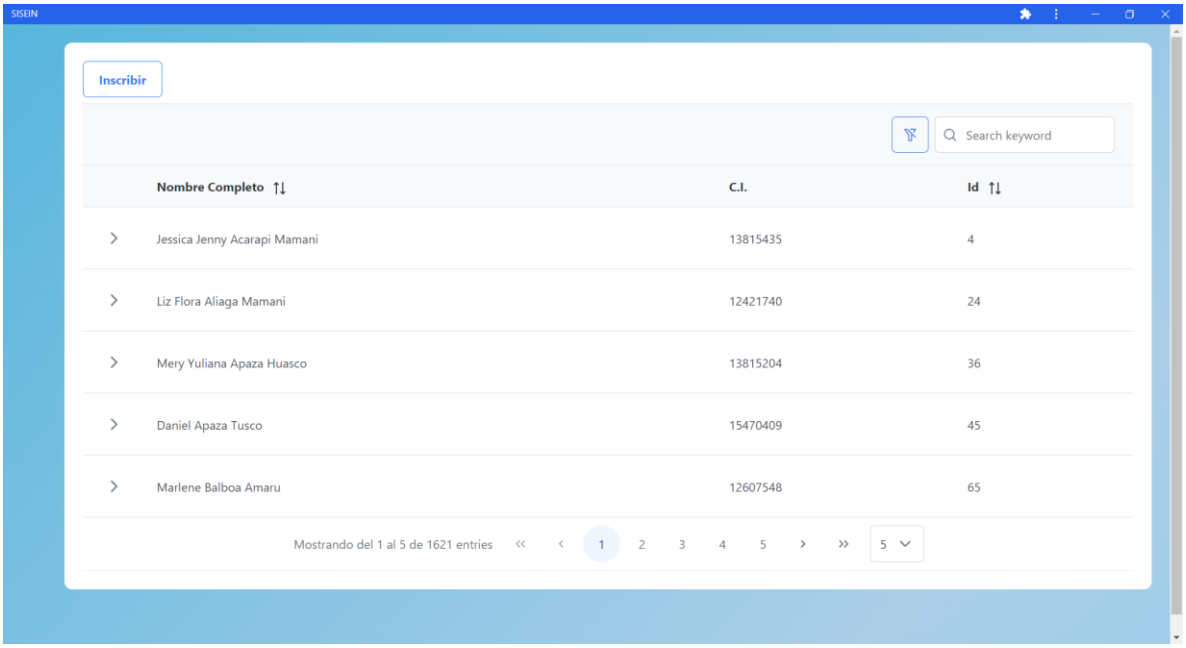
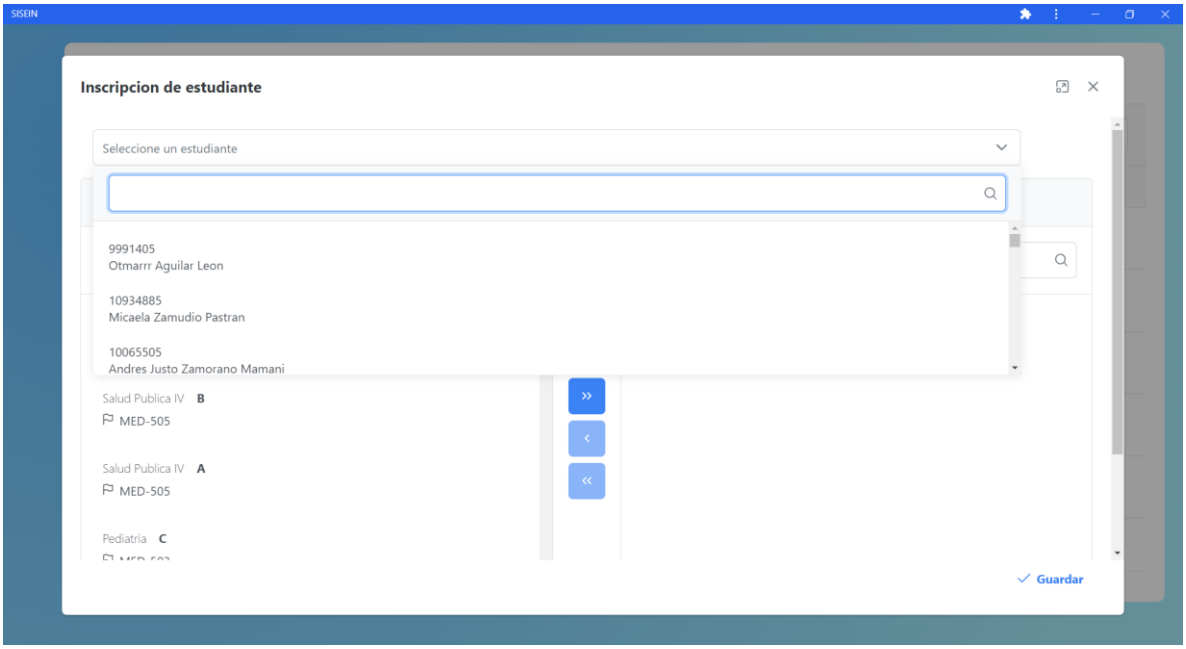


Figura 3.51

Inscripción de estudiante



3.4 Fase de pruebas

Esta fase permite verificar a la par con el cliente verificando si se cumplieron los requerimientos específicos en las historias de usuario. Para cada historia de usuario se elaboró una tarjeta de prueba de aceptación indicando el resultado de las tareas establecidas para cumplir con los requerimientos especificados en cada una.

Tabla 3.58

Tarjeta de Prueba de aceptación N° 1

N° Prueba de Aceptación: 1

N° Historia de usuario: 1

Descripción: El usuario accede al sistema a la pantalla de inicio de sesión e intenta acceder con sus credenciales de acceso,

Condiciones de ejecución:

- El usuario debe estar dado de alta por el administrador.
- El usuario debe contar con sus credenciales de acceso.

Pasos para de ejecución:

1. El usuario ingresa al sistema
2. El usuario llena los campos nombre de usuario y contraseña
3. Si las credenciales son correctas el sistema redirecciona a la pagina de inicio del sistema caso contrario no se ingresa vuelve a intentar el paso 2

Evaluación: Aprobada

Tabla 3.59

Tarjeta de Prueba de aceptación N° 2

N° Prueba de Aceptación: 2

N° Historia de usuario: 2

Descripción: El administrador crea un nuevo estudiante en el sistema

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos personales del estudiante
- Solo el administrador puede crear estudiantes

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de estudiante
3. Pulsa el botón nuevo en el módulo de estudiante
4. Introduce los datos del estudiante
5. Si el estudiante no está registrado se guardará en el sistema caso contrario saldrá una alerta con el mensaje “El usuario ya existe”
6. El estudiante aparece en la lista de estudiantes

Evaluación: Aprobada

Tabla 3.60

Tarjeta de Prueba de aceptación N° 3

N° Prueba de Aceptación: 3

N° Historia de usuario: 2

Descripción: El administrador modifica datos de un estudiante

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos personales del estudiante
- Solo el administrador puede editar estudiantes
- El estudiante debe esta registrado previamente

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de estudiante
3. Busca al estudiante que requiere una modificación de sus datos
4. Pusa el botón modificar en la sección de acciones
5. Introduce los datos del estudiante

6. Pulsa el botón guardar
7. Se guardan los cambios en los datos

Evaluación: Aprobada

Tabla 3.61

Tarjeta de Prueba de aceptación N° 4

N° Prueba de Aceptación: 4

N° Historia de usuario: 2

Descripción: El administrador elimina uno o varios estudiantes

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos personales del estudiante
- Solo el administrador puede eliminar estudiantes
- El estudiante debe está registrado previamente

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de estudiante
3. Busca y selecciona al estudiante/es que va a eliminar
4. Pusa el botón eliminar en la sección de acciones
5. Pulsa el botón eliminar
6. Se guardan los cambios en los datos

Evaluación: Aprobada

Tabla 3.62

Tarjeta de Prueba de aceptación N° 5

N° Prueba de Aceptación: 5

N° Historia de usuario: 3

Descripción: El administrador crea un nuevo docente en el sistema

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos personales del estudiante
- Solo el administrador puede crear docentes

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de docentes
3. Pulsa el botón nuevo en el módulo de docentes
4. Introduce los datos del docente
5. Si el docente no está registrado se guardará en el sistema caso contrario saldrá una alerta con el mensaje “El usuario ya existe”
6. El nuevo docente aparece en la lista de docentes

Evaluación: Aprobada

Tabla 3.63

Tarjeta de Prueba de aceptación N° 6

N° Prueba de Aceptación: 6

N° Historia de usuario: 3

Descripción: El administrador modifica datos de un docente

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos personales del estudiante
- Solo el administrador puede editar docentes
- El docente debe esta registrado previamente

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de docentes

3. Busca al docente que requiere una modificación de sus datos
4. Pusa el botón modificar en la sección de acciones
5. Introduce los datos del docente
6. Pulsa el botón guardar
7. Se guardan los cambios en los datos

Evaluación: Aprobada

Tabla 3.64

Tarjeta de Prueba de aceptación N° 7

N° Prueba de Aceptación: 7

N° Historia de usuario: 3

Descripción: El administrador elimina uno o varios docentes

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos personales del docente
- Solo el administrador puede eliminar estudiantes
- El docente debe esta registrado previamente

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de docente
3. Busca y selecciona al docente/es que va a eliminar
4. Pusa el botón eliminar en la sección de acciones
5. Pulsa el botón eliminar
6. Se guardan los cambios en los datos

Evaluación: Aprobada

Tabla 3.65

Tarjeta de Prueba de aceptación N° 8

N° Prueba de Aceptación: 8

N° Historia de usuario: 4

Descripción: El administrador crea nueva competencia

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos de la competencia

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de competencias
3. Pulsa en el botón nuevo en el módulo de competencias
4. Introduce los datos de la competencia
5. Pulsa en guardar
6. La nueva competencia se lista en las competencias

Evaluación: Aprobada

Tabla 3.66

Tarjeta de Prueba de aceptación N° 9

N° Prueba de Aceptación: 9

N° Historia de usuario: 4

Descripción: El administrador modifica una competencia

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los nuevos datos de la competencia

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de competencias

3. Busca la competencia que requiere modificación en los datos
4. Pulsa en el botón editar en el en apartado de acciones en el módulo de competencias
5. Introduce los datos de la competencia
6. Pulsa en guardar
7. La nueva competencia se lista en las competencias

Evaluación: Aprobada

Tabla 3.67

Tarjeta de Prueba de aceptación N° 10

N° Prueba de Aceptación: 10

N° Historia de usuario: 4

Descripción: El administrador elimina una o varias competencias

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos de la competencia que requiere eliminar

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de competencias
3. Busca y selecciona la competencia que requiere eliminar en los datos
4. Pulsa en el botón eliminar en el en apartado de acciones en el módulo de competencias
5. Se elimina la competencia

Evaluación: Aprobada

Tabla 3.68

Tarjeta de Prueba de aceptación N° 11

N° Prueba de Aceptación: 11

N° Historia de usuario: 5

Descripción: El administrador crea nueva asignatura

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos de la asignatura

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de asignaturas
3. Pulsa en el botón nuevo en el módulo de asignaturas
4. Introduce los datos de la asignatura
5. Pulsa en guardar
6. La nueva asignatura se lista en las asignaturas

Evaluación: Aprobada

Tabla 3.69

Tarjeta de Prueba de aceptación N° 12

N° Prueba de Aceptación: 12

N° Historia de usuario: 5

Descripción: El administrador modifica una asignatura

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los nuevos datos de la asignatura

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de asignaturas

3. Busca la competencia que requiere modificación en los datos
4. Pulsa en el botón editar en el en apartado de acciones en el módulo de asignaturas
5. Introduce los datos de la asignatura
6. Pulsa en guardar
7. La nueva asignatura se lista en las asignaturas

Evaluación: Aprobada

Tabla 3.70

Tarjeta de Prueba de aceptación N° 13

N° Prueba de Aceptación: 13

N° Historia de usuario: 5

Descripción: El administrador elimina una o varias asignaturas

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos de la asignatura que requiere eliminar

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de asignaturas
3. Busca y selecciona la asignatura que requiere eliminar en los datos
4. Pulsa en el botón eliminar en el en apartado de acciones en el módulo de asignaturas
5. Se elimina la asignatura

Evaluación: Aprobada

Tabla 3.71

Tarjeta de Prueba de aceptación N° 14

N° Prueba de Aceptación: 14

N° Historia de usuario: 5

Descripción: El administrador asigna competencia a una asignatura

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos de la asignatura
- Las competencias deben estar previamente creadas

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de asignaturas
3. Busca y selecciona la asignatura que requiere asignar las competencias
4. Pulsa en el botón asignar competencias en el en apartado de acciones en el módulo de asignaturas
5. Selecciona las competencias que desean designar a la asignatura
6. Pulsa el botón guardar

Evaluación: Aprobada

Tabla 3.72

Tarjeta de Prueba de aceptación N° 15

N° Prueba de Aceptación: 15

N° Historia de usuario: 6

Descripción: El administrador Elimina calificación

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos de la asignatura
- El administrador debe tener datos de la calificación eliminar
- Solo se puede eliminar la calificación de la asignatura si y solo si ningún estudiante fue calificado en esa instancia de la calificación

Pasos para de ejecución:

1. Ingresa al panel de administración

2. Ingresa al módulo de asignaturas
3. Busca y selecciona la asignatura donde requiere eliminar la calificación
4. Selecciona la calificación que desea eliminar
5. Se elimina la calificación

Evaluación: Aprobada

Tabla 3.73

Tarjeta de Prueba de aceptación N° 16

N° Prueba de Aceptación: 16

N° Historia de usuario: 7

Descripción: El docente crea una calificación de parcial en una asignatura

Condiciones de ejecución:

- El docente debe estar con una sesión iniciada
- El docente debe tener por lo menos una asignatura
- La asignatura debe tener inscritos a los estudiantes

Pasos para de ejecución:

1. Ingresa a las asignaturas
2. Selecciona la asignatura donde se desea crear la calificación parcial
3. Pulsa el botón crear parcial
4. Introduce la descripción y la calificación del parcial

Evaluación: Aprobada

Tabla 3.74

Tarjeta de Prueba de aceptación N° 17

N° Prueba de Aceptación: 17

N° Historia de usuario: 8

Descripción: El docente crea una calificación de practica en una asignatura

Condiciones de ejecución:

- El docente debe estar con una sesión iniciada
- El docente debe tener por lo menos una asignatura
- La asignatura debe tener inscritos a los estudiantes

Pasos para de ejecución:

5. Ingresa a las asignaturas
6. Selecciona la asignatura donde se desea crear la calificación practica
7. Pulsa el botón crear parcial
8. Introduce la descripción y la calificación del practica

Evaluación: Aprobada

Tabla 3.75

Tarjeta de Prueba de aceptación N° 18

N° Prueba de Aceptación: 18

N° Historia de usuario: 7,8

Descripción: El docente califica el parcial o practica de un estudiante (tanto como para parcial y practicas es el mismo procedimiento)

Condiciones de ejecución:

- El docente debe estar con una sesión iniciada
- El docente debe tener por lo menos una asignatura
- La asignatura debe tener inscritos a los estudiantes
- La calificación ya debe estar creada

Pasos para de ejecución:

1. Ingresa a las asignaturas
2. Selecciona la asignatura donde se desea crear la calificación parcial
3. Selecciona en botón calificar en la lista de los estudiantes
4. Selecciona la calificación que quiere calificar
5. Introduce la calificación obtenida por el estudiante

Evaluación: Aprobada

Tabla 3.76

Tarjeta de Prueba de aceptación N° 19

N° Prueba de Aceptación: 19

N° Historia de usuario: 9

Descripción: El docente adjudica la competencia al estudiante dependiendo de las particularidades de la asignatura.

Condiciones de ejecución:

- El docente debe estar con una sesión iniciada
- El docente debe tener por lo menos una asignatura
- La asignatura debe tener inscritos a los estudiantes
- Las competencias deben están designadas a la asignatura

Pasos para de ejecución:

1. Ingresa a las asignaturas
2. Selecciona la asignatura donde se desea crear la calificación parcial
3. Pulsa el botón Dar competencia en la lista de los estudiantes
4. Selecciona las una o varias competencias las cuales se adjudicarán al estudiante
5. Pulsa el botón guarda

Evaluación: Aprobada

Tabla 3.77

Tarjeta de Prueba de aceptación N° 20

N° Prueba de Aceptación: 20

N° Historia de usuario: 10

Descripción: El estudiante puede observar las sus calificaciones y competencias obtenidas durante la gestión en curso

Condiciones de ejecución:

- El estudiante debe estar con una sesión iniciada
- El estudiante debe tener por lo menos una asignatura inscrita

Pasos para de ejecución:

1. Ingresa a las calificaciones
2. Se despliega la lista de las calificaciones obtenidas
3. Ingresa a competencias
4. Se despliega la lista de las competencias adquiridas

Evaluación: Aprobada

Tabla 3.78

Tarjeta de Prueba de aceptación N° 21

N° Prueba de Aceptación: 21

N° Historia de usuario: 11

Descripción: El usuario cierra la sesión

Condiciones de ejecución:

- El usuario debe estar con una sesión iniciada

Pasos para de ejecución:

1. Pulsa el botón salir
2. Es redireccionado al módulo de inicio de sesión

Evaluación: Aprobada

Tabla 3.79

Tarjeta de Prueba de aceptación N° 22

N° Prueba de Aceptación: 22

N° Historia de usuario: 12

Descripción: El administrador inscribe a un estudiante a las asignaturas

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- Las asignaturas deben estar ya creadas
- La cuanta de estudiante ya debe estar creada

Pasos para de ejecución:

1. Ingresa a inscripciones
2. Pulsa el botón inscribir
3. Selecciona al estudiante que desea inscribir
4. Selecciona uno a varias asignaturas a las que desea inscribir al estudiante del paso 4
5. Pulsa Guardar

Evaluación: Aprobada

3.5 Fase de Lanzamiento

Ya cumpliendo con las fases anteriores se ha desarrollado un producto que cumple con las expectativas de los requerimientos en base a las historias de usuario se ha logrado desarrollar detalladamente un producto que cumple con las caracterizas principales de un buen software:

- Fiabilidad
- Eficiencia en el desempeño
- Usabilidad
- Seguridad
- Compatibilidad
- Mantenibilidad

Ya con el sistema desarrollado se procede con el llenado de los datos al sistema y el respectivo despliegue para el entregar al cliente el sistema de forma completa y funcional.

3.5.1 Despliegue

Para el despliegue se utilizará la plataforma Railway y Render ambas son plataformas de implementación donde puedes aprovisionar infraestructura, desarrollar con esa infraestructura localmente y luego implementarla en la nube.

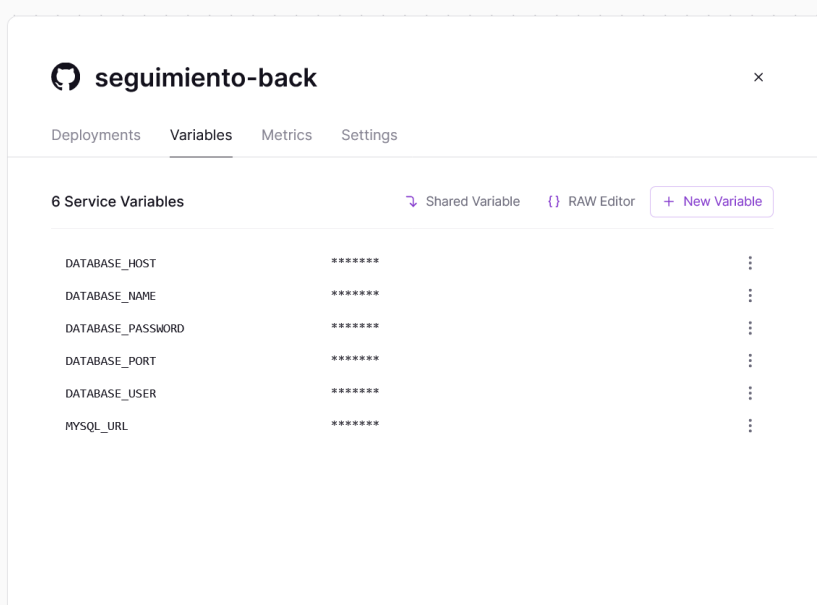
Al estar ambos en repositorio de GitHub repositorios frontend y backend se realiza el despliegue de manera continua en función de los cambios realizados.

Backend

El despliegue del backend se realizó con Railway en cual está alojado el servidor de NodeJs y la base de datos MySQL, bajo la siguiente configuración

Figura 3.52

Configuración de conexión Railway backend con base de datos



Se realizo la migración de los datos de:

- Estudiantes
- Docentes
- Asignatura
- Competencias
- Inscripciones
- Relación de competencia con asignaturas

Los cuales se obtuvieron de tablas Excel proporcionados por la carrera de medicina, en total se realiza la migración de 22955 registros.

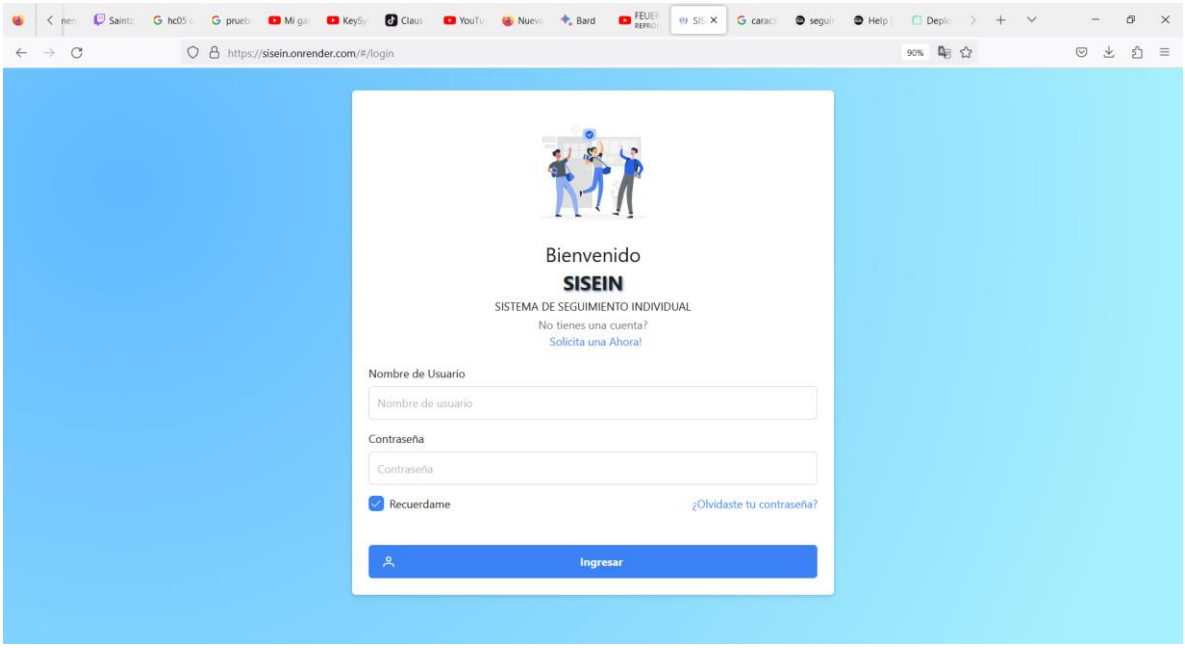
Frontend

El despliegue de frontend se realizó con render el cual proporciona el despliegue de página web el sitio se encuentra alojado en la siguiente url:

https://sisein.onrender.com

Figura 3.53

Sitio de alojamiento de la aplicación



Sin embargo una vez el sistema no requiera cambios se procederá a alojar el sistema web en la infraestructura del departamento SIE de la Universidad Pública de El Alto.

CALIDAD COSTO SEGURIDAD Y PRUEBAS

4.1 Introducción

El este capítulo se determina la calidad del software y desarrollo del software por lo tanto es uno de los puntos más importantes del proceso del desarrollo de software, también se realiza un énfasis en la seguridad y las pruebas para usando las herramientas determinadas por las normas correspondientes.

4.2 Calidad

La familia de normas ISO/EIC 25000 nos proporciona una serie de normas las cuales tienen por objetivo determinar la calidad del producto de software, en el presente proyecto parar la determinar la calidad obtenida del desarrollo de software se aplicará determinados controles los cuales se nombran en la norma ISO/EIC 25010, llamados requisitos y evaluación de calidad de productos de software SQuaRE.

4.2.1 Análisis de calidad del software

4.2.1.1 Funcionalidad

Es la capacidad del software de proveer las funciones para satisfacer las necesidades explicitas e implícitas, este atributo del sistema no puede ser calculado de forma directa, por esa razón se utiliza el cálculo de métrica de punto función, que contiene cinco características de información. Los valores de información son definidos de la siguiente forma:

Número de entradas de usuario: Cada entrada de usuario proporciona diferentes datos orientados a la aplicación. Las entradas se diferencian de las peticiones de forma separadas.

Número de salidas de usuarios: Cada salida proporciona al usuario información

orientada a la aplicación. Las salidas se refieren a informes, pantallas, mensajes de error entre otros.

Número de peticiones de usuario: Es una entrada interactiva que es producida por la respuesta del software inmediata en forma de salida interactiva. Las peticiones son contactadas de forma separada.

Numero de archivos: Un grupo lógico de datos que puede ser de una base de datos.

Numero de interfaces externas: Se cuenta todas las interfaces legibles que se utilizan para transmitir la información.

Para calcular los puntos función se emplea la siguiente formula:

$$PF = Cuenta\ Total * (0.65 + 0.01 * \sum F_i) \quad (1)$$

Donde:

$\boldsymbol{PF}$**:** Punto de función

$\boldsymbol{Cuenta\ Total}$**:** Sumatoria de los datos: N° de entradas, N° de salidas, N° de peticiones, N° de archivos y N° de interfaces externas.

$\mathbf{0.65}$**:** Confiabilidad del proyecto (1% - 100% => 0 a 1)

$\mathbf{0.01}$**:** Error mínimo aceptable de complejidad

$\sum \boldsymbol{F_i}$**:** Valores de ajuste de complejidad, donde $(1 \leq i \geq 14)$

Primeramente, se obtiene los datos: Cuenta Total y FI, analizando todas las interfaces desarrolladas del sistema se obtienen los siguientes datos:

Tabla 4.1

Tabla de obtención de datos de parámetros para el cálculo de fi

N°	Parámetros de Medida	Cantidad
1	Número de entradas de usuario	16
2	Número de salidas de usuario	12
3	Número de peticiones de usuario	26
4	Numero de archivos	12
5	Numero de interfaces externas	1

Una vez obtenida la información se procede a calcular la cuenta total con el factor de ponderación media que se muestra en la siguiente tabla:

Tabla 4.2

Tabla de ponderación

Parámetros de medida cantidad	Cantidad	Factor de Ponderación	Total
Número de entradas de usuario	16	4	64
Número de salidas de usuario	12	5	60
Número de peticiones de usuario	10	4	40
Numero de archivos	12	10	120
Numero de interfaces externas	1	7	7
		Cuenta total	291

Nota. el factor de ponderación se obtiene de la tabla de valores estándar (IFPUG) International Function Group, con una valoración de dificultad media.

Se obtiene los valores de ajuste de complejidad F_i en base a las respuestas de las siguientes preguntas que son evaluadas entre 0 a 5, se aplican en la siguiente tabla.

Tabla 4.3

Tabla de factor de complejidad

N°	Factores de complejidad	Sin Influencia	Incidencial	Moderado	Medio	Significativo	Esencial	F_i
		0	1	2	3	4	5	
1	¿Requiere el sistema copias de seguridad y de recuperación fiables?					X		4
2	¿Se requiere comunicación de datos?					X		4

3	¿Existen funciones de procesamiento distribuido?	X			3
4	¿Es crítico el rendimiento?		x		4
5	¿Se ejecutará el sistema con un sistema operativo existente y fuertemente utilizado?		X		4
6	¿Requiere el sistema entrada de datos interactiva?			X	5
7	Facilidad Operativa		X		4
8	¿Se actualiza los archivos maestros de forma interactiva?	x			3
9	¿Son complejas las entradas, las salidas, los archivos o las peticiones?		X		4
10	Procesamiento interno complejo	X			3
11	Diseño de código reutilizable		X		4
12	Facilidad de Instalación		X		4
13	¿Soporta múltiples instalaciones en diferentes sitios?		X		4
14	Facilidad de cambios			X	5
			Factor de ajuste de complejidad		**55**

Ya obtenidos los datos se procede a aplicar la ecuación 1, se pasa a remplazar los valores obtenidos en la tabla 4.3.

$$PF = 291 * (0.65 + 0.01 * 55)$$

$$\mathrm{PF}=349.2$$

Calculando el PF ideal se tiene:

$$PF_{ideal} = 291 * (0.65 + 0.01 * 70) \qquad (2)$$

$$PF_{ideal} = 392{,}85$$

Calculando la funcionalidad del sistema se tiene:

$$Funcionalidad = \frac{PF}{PFideal} * 100 \qquad (3)$$

$$Funcionalidad = \frac{349.2}{392.85} * 100$$

$$Funcionalidad = 88.88\%$$

Es decir, que la funcionalidad del sistema es de 89% para la satisfacción de las necesidades del usuario.

4.2.1.2 Fiabilidad

Métrica que mide la capacidad del software en su funcionamiento tomando en cuenta las fallas que puedan ocurrir en un tiempo determinado. Para el cálculo de fiabilidad de cada módulo se tiene la siguiente ecuación:

$$R(t) = e^{-\lambda t} \qquad (4)$$

Donde:

$R(t)$: Fiabilidad de un subsistema t.

λ: Tasa de constantes de fallo donde:

$$\lambda = N° \ de \ fallas \ de \ acceso \ / \ N° \ total \ de \ accesos \ al \ sistema$$

t: Periodo de operación de tiempo.

$e^{-\lambda t}$: Probabilidad de falla de un subsistema en el tiempo t.

Realizando las pruebas a cada módulo durante 4 horas, se obtienen los siguientes datos:

Tabla 4.4

Tabla para el cálculo de fiabilidad

N°	Módulo	$\boldsymbol{\lambda}$	$\boldsymbol{t}$	$\boldsymbol{R(t)}$
1	Módulo de Estudiantes	0,012	4 Hrs.	0,953133787
2	Módulo de Docentes	0,003	4 Hrs.	0,988071713
3	Módulo de Asignaturas	0,01	4 Hrs.	0,960789439
4	Módulo de competencias	0,001	4 Hrs.	0,996007989
5	Módulo de Inscripciones	0,024	4 Hrs.	0,908464016
6	Módulo de inicio de Sesión	0,032	4 Hrs.	0,879853379
7	Módulo de calificaciones	0,02	4 Hrs.	0,923116346
8	Módulo de consulta de calificaciones y Competencias	0,019	4 Hrs.	0,926816207
9	Módulo de asignación de competencias	0,022	4 Hrs.	0,915760877

Para el cálculo de la fiabilidad del sistema completo se tiene la siguiente ecuación:

$$Fiabilidad = R_s * R_p \qquad (5)$$

Donde: $R_s = R_1 =$ 0,953133787 (6)

$$Rp = \frac{\sum_2^5 (R_i * P_i)}{\sum_2^5 P_i} \quad (6)$$

Para la ecuación de R_p (6), se asigna a P_i el valor 1, ya que se refiere a la participación del equipo para el desarrollo del sistema, como la participación fue de 100% se le asigna el valor 1.

Reemplazando la ecuación de (6) se tiene:

$$Rp = \frac{\sum_2^5 (R_i * P_i)}{\sum_2^5 P_i} = \frac{0{,}9881 + 0{,}961 + 0{,}996 + 0{,}908 + 0{,}88 + 0{,}923 + 0{,}927 + 0{,}9158}{8}$$

$$Rp = 0{,}937359996$$

Reemplazando datos en la ecuación (5) se obtiene:

$$Fiabilidad = 0{,}953133787 * 0.71 = 0.66 * 100 = 66\%$$

$$Fiabilidad = 0{,}983133787 * 0{,}937359996 = 0{,}8934249 = 89\%\ de\ fiabilidad$$

De lo cual se puede decir que existe un 11% de probabilidad de que el sistema presente algún fallo cuando se exceda un tiempo de uso continuo, debido a que puedan existir fallas con la conexión del sistema a la base de datos, conexión del cliente al sistema, uso incorrecto del sistema por parte del usuario, errores en la entrada de datos.

4.2.1.3 Usabilidad

La usabilidad es la capacidad del software de ser entendido, aprendido, y usado de forma fácil y atractiva. Para determinar el porcentaje de la usabilidad del sistema se optó por realizar una encuesta mediante la herramienta Google forms a 8 personas. La siguiente tabla nos muestra los resultados de la encuesta que se realizó:

Tabla 4.5

Tabla de encuestas sobre usabilidad

N°	Preguntas	Respuestas		% de si
		Si	No	
1	¿Aprendió rápido a usar el sistema?	7	1	87,5
2	¿Las pantallas que vio fueron de su agrado?	8	0	100

3	¿Las pantallas que vio fueron fáciles de comprender?	8	0	100
4	¿El sistema responde rápido a sus solicitudes?	6	2	75
5	¿El sistema le facilitara el trabajo?	7	1	87,5
6	¿El sistema reduce su tiempo de trabajo?	8	0	100
7	¿Es fácil navegar por las distintas opciones?	8	0	100
8	¿Las operaciones que se realizan no son complicadas?	7	1	87,5
9	¿El sistema le proporciono las respuestas requeridas?	6	2	75
10	¿El sistema no presento errores?	8	0	100
	Promedio			91,25

Se determina que el sistema tienes 91,25% de usabilidad

4.2.1.4 Mantenibilidad

La mantenibilidad se refiere a la cualidad del software para ser modificado, corregido o mejorado. Para obtener la calidad de mantenimiento se utilizará el índice de madurez (IMS), para determinar la estabilidad del producto. La ecuación de índice de madurez es la siguiente:

$$ISM = \frac{Mt-(F_a+F_b+F_c)}{Mt} \quad (7)$$

Donde:

$\boldsymbol{M_t}$**:** N° de módulos en la versión actual.

$\boldsymbol{F_a}$**:** N° de módulos en la versión actual que se han cambiado.

$\boldsymbol{F_b}$**:** N° de módulos en la versión actual que se han añadido.

$\boldsymbol{F_c}$**:** N° de módulos en la versión anterior que se han borrado en la versión actual.

Recopilando la información necesaria, se obtuvo que:

$\boldsymbol{M_t} = 9$ por el número de módulos actuales en el sistema.

$\boldsymbol{F_a, F_b, F_c} = 0$ por qué no se cuenta con estos valores.

Con los valores obtenidos se reemplaza en la ecuación (7) de IMS:

$$ISM = \frac{9 - (0)}{9} = 1$$

$$ISM = 1 * 100\%$$

Se concluye que el sistema tiene un índice de madurez del 100%.

4.2.1.5 Portabilidad

Dentro del análisis de portabilidad se desarrolló las siguientes características:

- Capacidad para ser reemplazado
- Adaptabilidad

La primera característica que refiere a la capacidad que tiene el software para ser llevado de un entorno a otro, tomando en cuenta la facilidad de instalación, ajuste y adaptación al cambio. Se cuenta con la siguiente fórmula para obtener el grado de portabilidad:

$$GP = 1 - \frac{ET}{ER}$$

Donde:

$\boldsymbol{GP}$: Grado de portabilidad. Si:

$\boldsymbol{GP} > \mathbf{0}$, la portabilidad es más rentable que el re-desarrollo.

$\boldsymbol{GP} < \mathbf{0}$, el re-desarrollo es más rentable que la portabilidad.

$\boldsymbol{GP} = \mathbf{1}$, la portabilidad es perfecta.

$\boldsymbol{ET}$: Recursos necesarios para llevar el sistema a otro entorno.

$\boldsymbol{ER}$: Recursos necesarios para crear el sistema en el entorno residente.

Los recursos que se requiere para trasladar el sistema a otro entorno son:

- Para el despliegue del frontend se requiere un servidor web
- Para el despliegue del backend se requiere un servidor Node
- La Base de datos se requiere una instancia de Mysql

Considerado los aspectos citados se tiene: ET=3

Para crear el sistema en el entorno residente se necesita contar con un equipo que tenga sistema operativo de (Windows, Linux o Mac OS) en los cuales se requiere tener:

- Editor de código
- Instancia de Base de datos MySQL
- Navegador web
- Tener instalado Node (npm)
- Angular CLI
- Nest CLI
- Gestor de Base de datos
- Typeorm
- PrimeNG
- Primeflex

Considerado los aspectos citados se tiene: ER=3

Tomando en cuenta con la información descrita anteriormente, se calcula el grado de portabilidad.

$$GP = 1 - \frac{3}{10}$$

$$GP = 1 - 0.3$$

$$GP = 70\%$$

Con el resultado obtenido se concluye que el sistema tiene un grado de portabilidad en capacidad para ser reemplazado de 70%.

Por otra parte, la portabilidad de igual manera, incluye la Adaptabilidad la cual es la capacidad de ser adaptado de forma efectiva y eficiente a diferentes entornos determinados de hardware, software, operacionales o de uso, el software esta desarrollado bajo el concepto Mobile First de esta manera en la parte de portabilidad se tiene 100% de adaptabilidad

Por lo tanto, sacando el promedio de ambos se tiene 85%.

4.2.1.6 Calidad Global

Una vez calculado los porcentajes de los diferentes atributos que el sistema tiene según lo propuesto por el estándar de calidad ISO/IEC 25010, se procedió a calcular la calidad global del sistema, la cual se visualiza en la siguiente tabla:

Tabla 4.6

Tabla de cálculo de calidad Global de sistema

Métrica	Valores en %
Funcionalidad	89
Fiabilidad	89
Usabilidad	91,25
Mantenibilidad	100
Portabilidad	85
Calidad Global	90,85

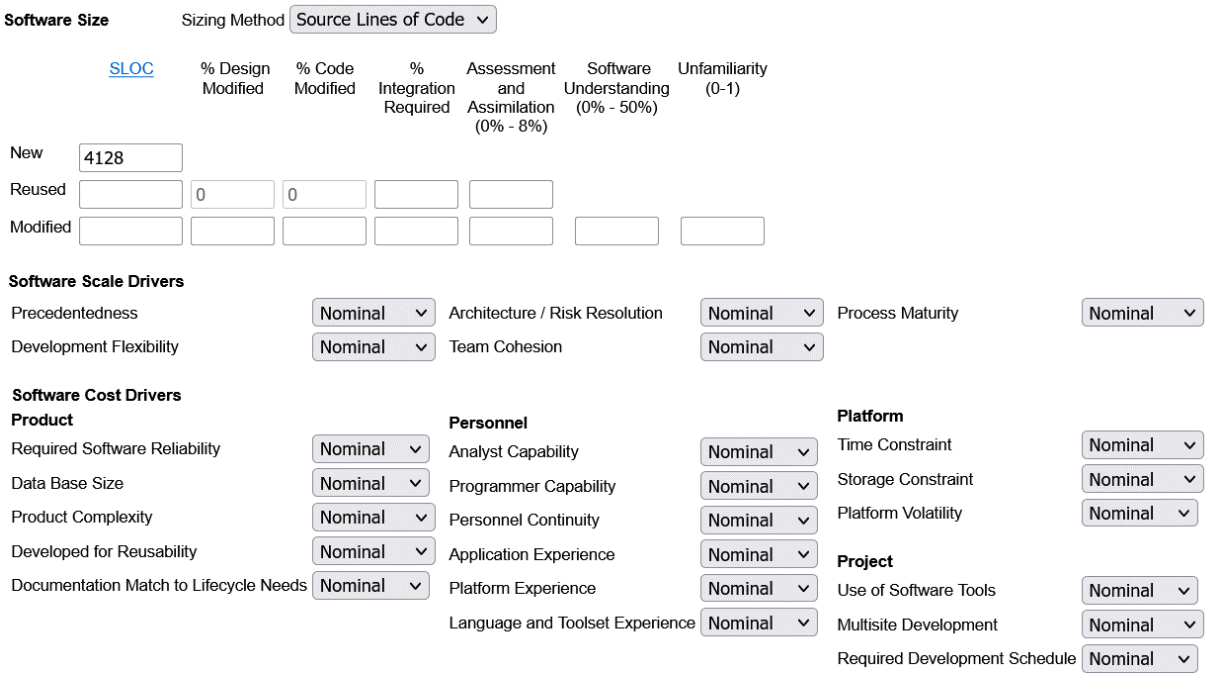
Con el valor obtenido se determina que la calidad del sistema es de un 90,85%

4.3 Costos

Para la obtención de la estimación de costos se utilizó del modelo constructivo de costos COCOMO II el cual proporciona una serie de parámetros dependiendo de las características de software desarrollado, se usó la página web: http://softwarecost.org/tools/COCOMO/, la cual nos proporciona una forma sencilla de hacer la estimación de costos en base a COCOMO II.

Figura 4.1

Costos con la página web COCOMO II



Nota. Se pone la cantidad de líneas de código que son 4128 es la suma de líneas de código escritas tanto en el backend y frontend, también se introducen los datos para el cálculo de los atributos Multiplicador de Esfuerzo.

Luego de introducir los datos requeridos nos da los resultados sobre los datos de: Esfuerzo, Tiempo y Costos.

Figura 4.2

Resultados COCOMO II

Nota. Se introduce el sueldo mínimo en Bolivia el cual es: Bs 2362 en dólares 342.

Entonces una vez obtenidos los resultados se determina los siguiente:

- El Esfuerzo es de 14 Personas por mes
- Tiempo de desarrollo de 9 meses y medio
- Costo total $ 4780 o expresado en Moneda Nacional Bs 33.029

4.3.1 Costo de elaboración de proyecto

A continuación, se procede a describir los recursos que se invirtieron en la elaboración del proyecto.

Tabla 4.7

Tabla de costo de elaboración de proyecto

Recurso	Costo (Bs)
Material de escritorio	200
Investigación de Proyecto	400
Internet	600
Otros	200
Total	1.400

4.3.2 Costo total del sistema

El costo total del sistema se obtiene de la sumatoria del costo de desarrollo y el costo de elaboración del proyecto.

Tabla 4.8

Tabla de costo total del sistema usando COCOMO

Detalle	**Costo (Bs)**
Costo de desarrollo	33.029
Costo de elaboración de proyecto	1.400
Total	34.429

Sin embargo, COCOMO solo nos da una estimación del costo por lo cual basándose en los recursos realmente usados para el desarrollo se aplica otra forma de analizar los costos, dependido de las siguientes entradas:

- El desarrollo completo de sistema tomo 6 meses.
- El sueldo promedio de un programador junior en Bolivia es de 5,125 Bs. (el salario promedio de un programador junior ya incluye el multiplicador de esfuerzo por el salario mínimo de Nacional).
- Un solo programador en el Proyecto.

Tomando en cuenta los anteriores parámetros se obtiene el resultado de costo de desarrollo de software de 30.750 Bs, por lo tanto, en relación al anterior costo estimado con COCOMO no se tiene mucha diferencia.

4.4 Seguridad

Por la parte de seguridad al ser uno de los aspectos más importantes para el correcto manejo de la información dentro del sistema web desarrollado se tomó en cuenta los determinados controles nombrados dentro de la ISO 27001:2013

4.4.1 Acceso controlado

El acceso al sistema está determinado por un sistema de roles los cuales son:

- Estudiante
- Docente
- Administrador

Cada usuario dependiendo de su rol tiene el acceso a determinadores módulos de interés dependiendo del tipo de usuario que acceda al sistema.

Al momento de iniciar la sesión se genera un token JWT el cual expira después de cierto tiempo, iniciada la sesión dicho token contiene información del usuario como ser sus datos personales y su rol, el mismo token se usa para las peticiones http al servidor backend en cual verifica si el token está firmado por el mismo backend para la verificación del usuario.

4.4.2 Autorización

El usuario dependiendo de su rol que posea tendrá acceso a delimitada información, por ejemplo, en el caso del estudiante un estudiante solo puede ver la información concerniente a el mismo puede solicitar datos como ser: materia en las que está inscrito, calificaciones obtenidas y competencias adquiridas, el rol de estudiante solo tiene acceso a esa información.

Los docentes solo pueden solicitar información de las asignaturas en las cuales están inscritos, las lista de sus estudiantes, calificaciones realizadas y competencias adjudicadas.

El sistema determina la clasificación de la información dependiendo del rol del usuario como indica la iso 27001:2013 nos dice que se debe clasificar la información de la siguiente manera:

- Confidencial: cuando el nivel de confidencialidad de la información se incremente.
- Restringido: para niveles medios de confidencialidad.

- Uso interno: información con un nivel bajo de confidencialidad.
- Público: cuando todas las personas pueden ver la información.

4.4.3 Seguridad física

El sistema en la fase de producción al estará alojado en los servidore de unidad SIE de la Universidad pública de el alto, dicha unidad nos garantiza que nos proporciona los ambientes que cumplen con los requerimientos de la iso 27001:2013 los cuales son:

- Aseguran el control de acceso a las áreas seguras solo está disponible a personas autorizadas.
- Establece un perímetro de seguridad, instalar controles de acceso a una instalación e implantar medidas contra inundaciones e incendios.
- Controla y registra cada una de las visitas.
- En oficinas y despachos, la información crítica estará alejada de zonas de acceso público, fotocopiadoras y faxes estarán apartadas del área de seguridad y se instalarán alarmas para detectar intrusos.
- El trabajo en las áreas seguras y sus actividades solo las conocerán el personal necesario, y éstas deberán estar cerradas y vigiladas.
- El material que entre a la organización se deber inspeccionar y registrar antes de introducirlo en la organización.

En una entrevista se aclaró que cumplen con las buenas prácticas de seguridad dentro de la iso 27001: 2013

4.4.4 Criptografía

La información del usuario esta encriptada en el token JWT por una firma digital con SHA-256 de esta manera se garantiza la seguridad de la información del usuario, de la misma manera al momento de crear un usuario las contraseñas son encriptadas por la librería Bycript el cual lleva incorporado un valor llamado salt, que es un fragmento aleatorio que se usará para generar el hash asociado a la contraseña, y se guardará junto con ella en la base de datos. Así se evita que dos contraseñas iguales generen

el mismo hash y los problemas que ello conlleva, por ejemplo, ataque por fuerza bruta a todas las contraseñas del sistema a la vez.

4.5 Pruebas

Para realizar las pruebas del funcionamiento del sistema, se realizó, las pruebas unitarias en cada iteración de los módulos las cuales fueron descritas anteriormente. Además, se realizará la prueba de caja blanca y negra para la autentificación ya que es la parte primordial para el acceso al sistema.

4.5.1 Caja Blanca.

Mediante la técnica del camino básico se diseñará la prueba de caja blanca respecto a la creación de usuario y autentificación del usuario.

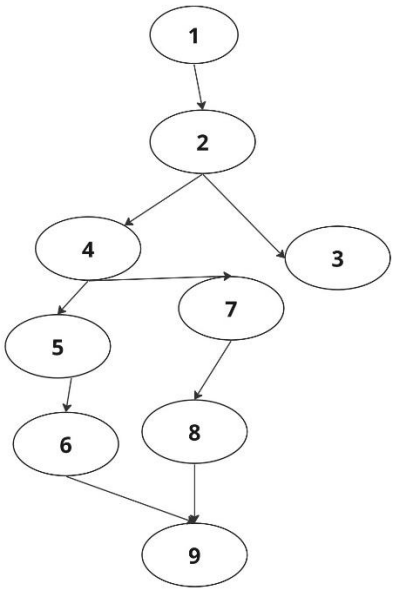
En la Figura 4.3 se realiza el etiquetado del código del acceso al sistema para posteriormente realizar el grafo correspondiente como se muestra en Figura 4.4 teniendo como complejidad ciclomática igual a 3.

Figura 4.3

Fragmento de código creación de usuarios modulo de usuarios

Figura 4.4

Grafo del código de Creación de usuario



Casos de prueba

Tabla 4.9

Tabla de prueba de nodos caso creación de usuario

N°	Camino independiente	Solicitud	Respuesta
1	1-2-4-5-6-9	{ "rol": 3, "nombres": "Otmar", "apellidoPaterno": "Aguilar", "apellidoMaterno": "Leon", "email": "ootmar1998@gmail.com", "telefono": "68103468", "direccion": "string", "ci": 9991405, "fnacimiento": "1998-02-10" }	Usuario estudiante creado

2	1-2-4-7-8-9	{
		"rol": 1,
		"nombres": "Otmar",
		"apellidoPaterno": "Aguilar",
		"apellidoMaterno": "Leon",
		"email": "ootmar1998@gmail.com",
		"telefono": "68103468",
		"direccion": "string",
		"ci": 9991405,
		"fnacimiento": "1998-02-10"
		}
			Usuario docente Creado

3	1-2-3	{
		"rol": 2,
		"nombres": "Otmar",
		"apellidoPaterno": "Aguilar",
		"apellidoMaterno": "Leon",
		"email": "ootmar1998@gmail.com",
		"telefono": "68103468",
		"direccion": "string",
		"ci": 9991405,
		"fnacimiento": "1998-02-10"
		}
			Alerta el usuario ya existe

Nota. Utilizando la técnica del camino básico en la creacion de usuario se verifica: Todas las aristas intervinientes en el grafo y las condiciones de evaluación de verdadero y falso.

Figura 4.5

Fragmento de código modulo autenticación de usuarios

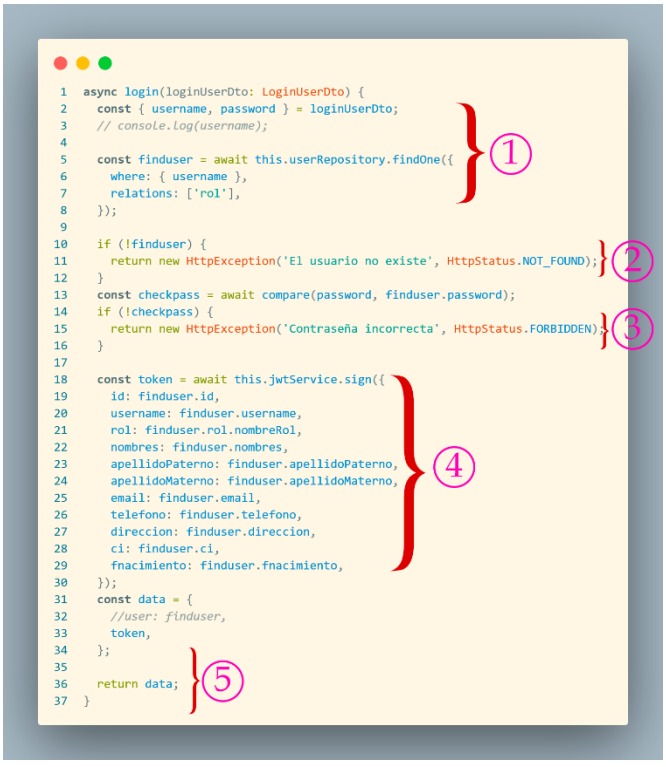


Figura 4.6

Grafo del código de autenticación de usuario

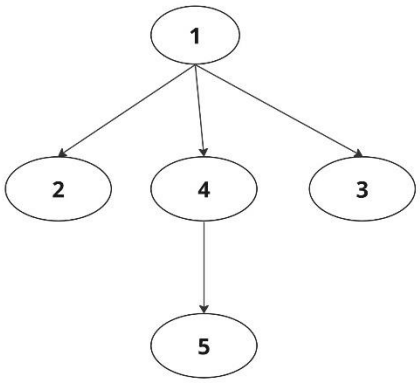


Tabla 4.10

Tabla de prueba de nodos caso autenticación de usuario

N°	Camino independiente	Solicitud	Respuesta
1	1-2	{ "username": "asd_1234", "password": "1234" }	El usuario no existe
2	1-4-5	{ "username": "Otmar_9991405", "password": "9991405" }	Ingresa correctamente
3	1-3	{ "username": "Otmar_9991405", "password": "99911405" }	Contraseña incorrecta

Nota. Utilizando la técnica del camino básico en la autentificación de usuario se verifica: Todas las aristas intervinientes en el grafo y las condiciones de evaluación de verdadero y falso.

4.5.2 Caja Negra

La prueba de caja negra se refiere a las pruebas que se llevan a cabo sobre la interfaz de usuario. Se pretende demostrar que las funciones del software son operativas, que las entradas se aceptan de forma adecuada y que se produce un resultado correcto.

Estas pruebas se pueden realizar para verificar el funcionamiento correcto de cada uno de los módulos del sistema. En el presente proyecto las pruebas de caja negra se la realizo a todos los módulos sin embargo se muestran las más concurrentes: inicio de sesión y creación estudiantes.

Figura 4.7

Login de sistema

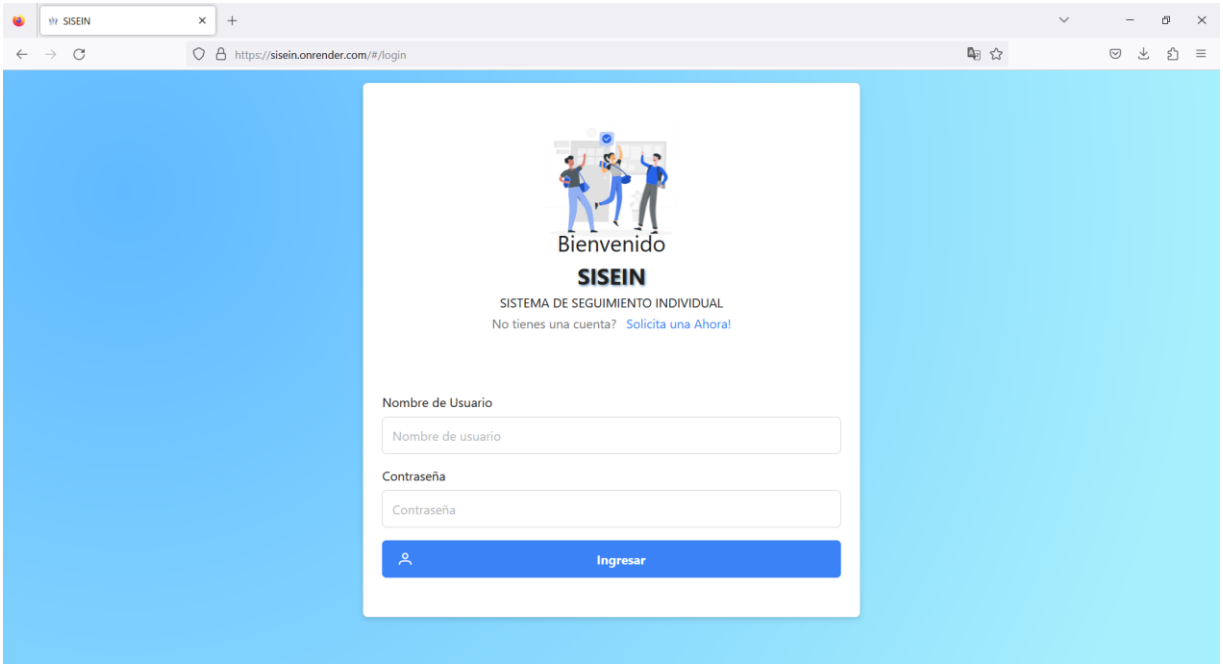


Tabla 4.11

Valores caso inicio de sesión

Campos	Entrada Valida	Entrada invalida
Usuario	Cadena de texto	Espacios en blanco.
Contraseña	Cadena de texto	Espacios en blanco.

Tabla 4.12

Prueba de caja negra inicio de sesión

Entradas	Salidas	Resultado
Otmar_9991405 ********	“Bienvenido al sistema”	El sistema valida que no se ingresen datos en blanco Al introducir datos validos el sistema concede al acceso al mismo

Figura 4.8

Formulario de creación de usuario estudiante

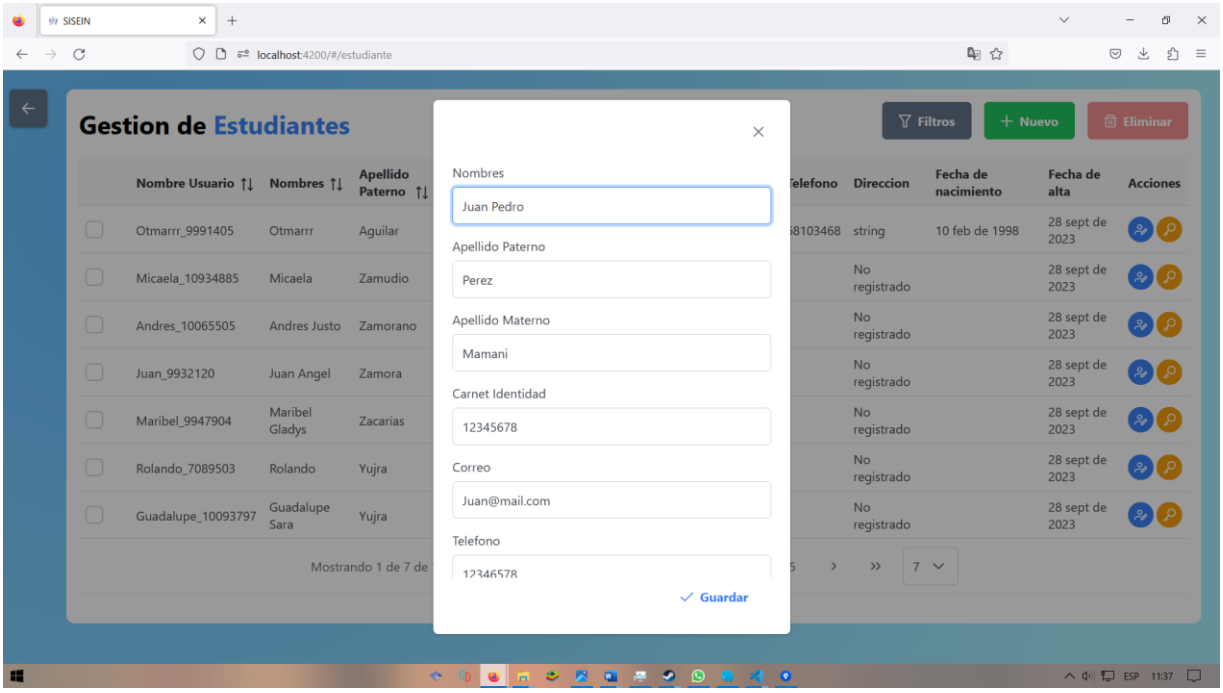


Tabla 4.13

Campos para creación de usuario

Campos	Entrada Valida	Entrada invalida
Nombres	Cadena de texto	Espacios en blanco. Números
Apellido paterno	Cadena de texto	Espacios en blanco. Números
Apellido Materno	Cadena de texto	Espacios en blanco. Números
Carnet Identidad	Cadena de números	Espacios en blanco. Caracteres alfabéticos
Correo	Correo electrónico	Correo invalido que no contenga (@)
Teléfono	Cadena de números	Espacios en blanco. Caracteres alfabéticos
Dirección	Cadena de texto	Ninguna
Fecha de nacimiento	Fecha valida	Espacios en blanco. Caracteres alfabéticos

Tabla 4.14

Prueba de caja negra creación de usuario

Entradas	Salidas	Resultado
Juan Perez Mamani 12345678	“Usuario Creado Correctamente”	El usuario se creó y se guarda en la base de datos y se lo

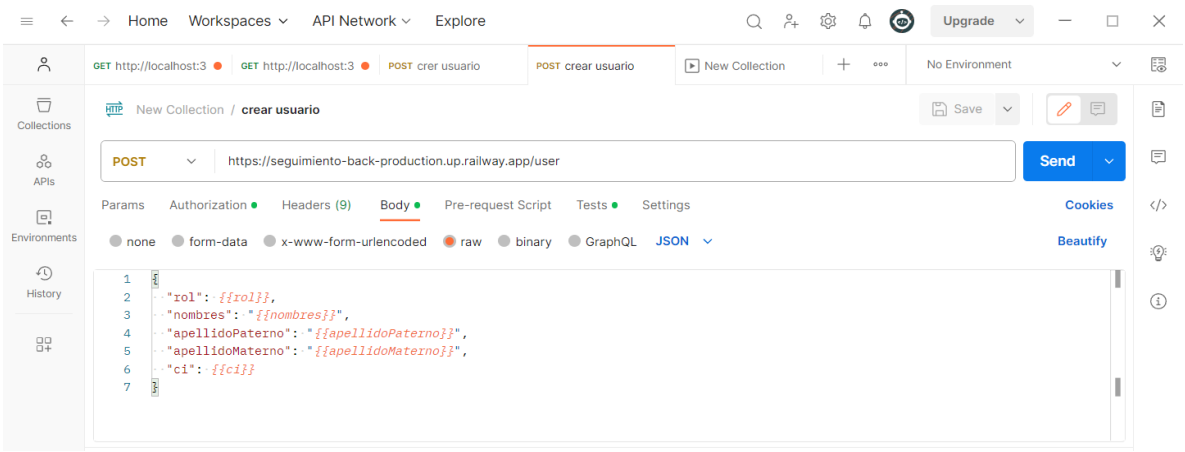
Juan@mail.com 12345678 C/ vereda n°100 10/02/1998		visualiza en tabla de usuarios
Todos los espacios en blanco	Nombres es requerido Apellido paterno es requerido Apellido Materno es requerido Carnet Identidad es requerido Correo es requerido Teléfono es requerido Dirección es requerido Fecha de nacimiento es requerido	No se realiza ninguna acción ya que no hay datos validos

4.5.3 Pruebas de Estrés

Probar un sistema enfatizando aspectos como la robustez, disponibilidad, manejo de errores y otros. Bajo una carga pesada, el objetivo de estas pruebas es de asegurar que el sistema no colapsara cuando disponga de pocos recursos o exista una gran concurrencia de usuarios, para el mismo recurrimos a la aplicación Postman la cual nos permite realizar una colección de solicitudes a la vez y ver la respuesta del servidor.

Figura 4.9

Solicitud al servidor endpoint creación de usuario

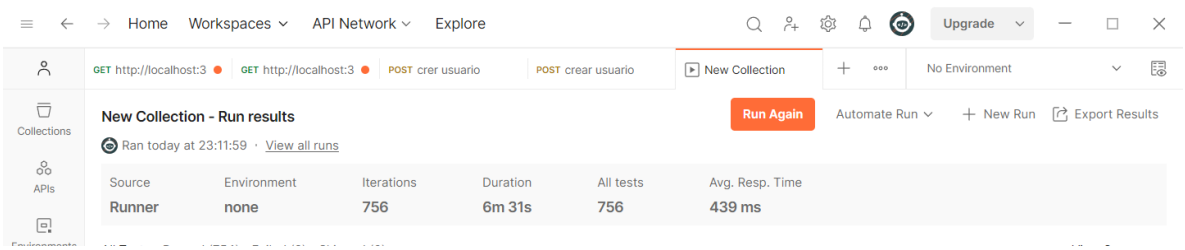


Nota. la solicitud al servidor se realizó con una colección de datos en formato JSON

Una vez creada la solicitud se espera a la respuesta del servidor, se realizó 756 solicitudes con un promedio de respuesta de 439 ms y una duración total de 6 minutos y 30 segundos

Figura 4.10

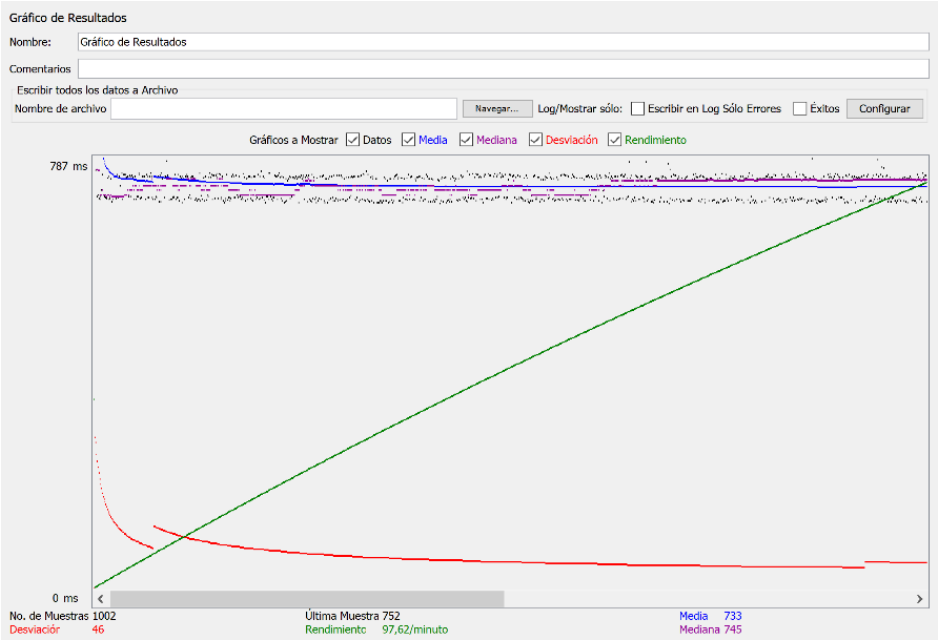
Respuesta la prueba del servidor al endpoint de creación de usuarios



De igual manera se realizó con pruebas con la herramienta JMeter con 1000 solicitudes al largo de 100 segundos dicha prueba se repitió 10 veces obteniendo los siguientes resultados los cuales muestran en las siguientes figuras.

Figura 4.11

Grafico de respuesta con JMeter



Nota. En el test se obtuvo un rendimiento promedio de 97,62/minutos

Figura 4.12

Respuestas a solicitud con JMeter

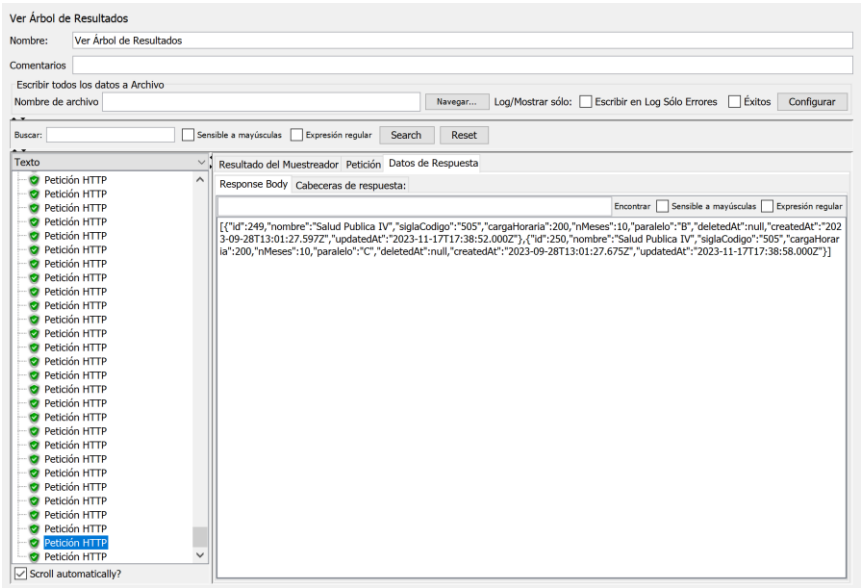
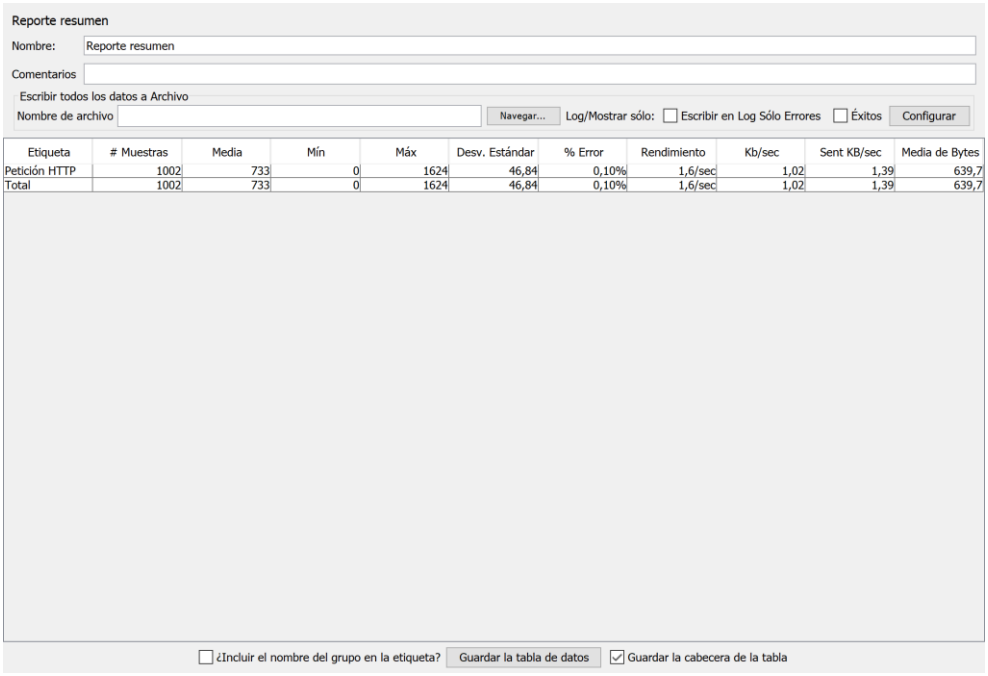


Figura 4.13

Sumario de Respuestas con JMeter



Nota. Se muestran los resultados obtenidos durante las pruebas de estres

CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

Se logro desarrollar e implementar un sistema de información en el cual se realiza el seguimiento a cada uno de los estudiantes tanto de sus calificaciones como de sus competencias así de esta manera hacer la verificación de su perfil profesional de una manera confiable oportuna y segura para la toma de decisiones.

De la misma en repuesta a nuestros objetivos específicos:

- Se logro hacer un análisis de la situación en la que se encontraban la gestión de la información de la calificaciones y competencias.
- Se logro brindar información de manera oportuna a docentes y administrativos sobre las competencias adquiridas de cada estudiante.
- Se sistematizo el proceso de generación de información de los estudiantes acerca de sus competencias y calificaciones de manera clara y concisa.
- Se diseño una base de datos la cual haga un manejo de la información a largo plazo de manera segura.
- Se creo una interfaz sencilla e intuitiva para la accesibilidad de los usuarios tanto docentes, estudiantes y administradores.

Por último, se concluye que todos los objetivos propuestos en el presente proyecto fueron alcanzados con lo cual se podrá obtener una gestión de la información de docentes, estudiantes, calificaciones y competencias.

5.2 Recomendaciones

A la culminación del presente proyecto se efectúan las siguientes recomendaciones:

- Se recomienda a los usuarios que interactúan con el sistema (usuarios registrados), que los datos de ingreso al sistema sean de conocimiento meramente personal, además de contar con una contraseña segura y no detectable para usuarios no autorizados al sistema, además de cambiar periódicamente su contraseña para una amplia seguridad.
- Realizar las respectivas copias de seguridad de la base de datos para el resguardo de la información.
- Hacer un seguimiento del sistema para corroborar el correcto funcionamiento en todo momento.
- Realizar reuniones frecuentes para el consultar sobre las experiencias de los usuarios con el sistema para recibir retroalimentación y añadir nuevas características o correcciones del sistema.

BIBLIOGRAFÍA

KeepCoding Team. (18 de Marzo de 2022). *¿Qué es MariaDB?* . Obtenido de https://keepcoding.io/blog/que-es-mariadb/

Lucid Software Inc. (s.f.). *Qué es el lenguaje unificado de modelado (UML)*. Obtenido de https://www.lucidchart.com/pages/es/que-es-el-lenguaje-unificado-de-modelado-uml

Acevedo Quispe, Y. L. (2018). Implementación de un sistema web para la mejora del proceso administrativo académico de la institución educativa “Wari-Vilca”- Huayucachi. *Para optar el título profesional de ingeniera de sistemas.* Universidad nacional del centro del Perú, Huancayo, Peru.

ACMS, G. (23 de Mayo de 2022). *¿Qué es ISO 25000? Certificación de Calidad de Software*. Obtenido de https://www.grupoacms.com/blog/iso-25000-certificacion-de-la-calidad-del-software

Addappto. (21 de Agosto de 2015). *¿Qué es un sistema web?* Obtenido de http://www.addappto.com/que-es-un-sistema-web/

Advisera. (s.f.). *¿Qué es norma ISO 27001?* . Obtenido de https://advisera.com/27001academy/es/que-es-iso-27001/

Alarcon Aroyo, O. (2017). Sistema web para el control y seguimiento de kárdex administrativo Caso: postgrado en Informática. *Para optar al título de licenciatura en informática.* Universidad Mayor De San Andrés, La paz, Bolivia.

Angular. (s.f.). *Introduction to the Angular Docs*. Obtenido de https://angular.io/docs

Angular.io. (28 de Febrero de 2022). *¿Qué es Angular?* Obtenido de https://angular.io/guide/what-is-angular

argentina.gob.ar. (s.f.). Obtenido de https://www.argentina.gob.ar/sites/default/files/2._perfil_profesional.pdf

Atlassian. (2022). *Los distintos tipos de pruebas en software | Atlassian*. Obtenido de https://www.atlassian.com/es/continuous-delivery/software-testing/types-of-software-testing

ATURA. (26 de Agosto de 2021). *Sistemas web y sus ventajas*. Obtenido de https://www.atura.mx/blog/sistemas-web-y-sus-ventajas

Barcelona Geeks. (5 de Julio de 2022). *Ventajas y desventajas de TypeScript sobre JavaScript*. Obtenido de https://barcelonageeks.com/ventajas-y-desventajas-de-typescript-sobre-javascript/

Boada Oriols, M., & Gómez Gutiérrez, J. A. (2019). *El gran libro de Angular.* México: Alfaomega Grupo Editor, S.A. de C.V.

Calamani Salas, J. M. (2020). “Sistema de información para el registro y seguimiento académico” caso: academia de formación profesional "Bela". *Para optar al título de Licenciatura en Ingeniería de Sistemas.* Universidad Publica De El Alto, El Alto, Bolivia.

Código Facilito. (3 de Junio de 2018). *Qué es TypeScript*. Obtenido de https://codigofacilito.com/articulos/typescript

Concepto Definición. (4 de Abril de 2021). *Seguimiento*. Obtenido de https://conceptodefinicion.de/seguimiento/

Cristian. (21 de Abril de 2018). *Definiciones de Palabras*. Obtenido de https://definicionesdepalabras.com/definicion-de-web/

DEVOPS LATAM. (15 de Abril de 2021). *Devops Latam*. Obtenido de https://devopslatam.com/15-metodos-de-prueba-que-todos-los-desarrolladores-deben-conocer/

Dorado Gómez, M. A. (2019). Plataforma GITLAB como Estrategia de Aprendizaje Colaborativo en el rendimiento académico de los estudiantes de la Carrera de Sistemas Informáticos del I.T.M.Q.S.C. Sede Central. *Tesis de Maestría para optar el grado académico de magister scientiarum en educación superior.* Universidad Mayor de San Andrés, La Paz, Bolivia.

Education IBM Cloud. (27 de Agosto de 2019). *IBM*. Obtenido de https://www.ibm.com/es-es/cloud/learn/database-security

educaweb. (s.f.). *¿Qué son las competencias profesionales?* Obtenido de https://www.educaweb.com/contenidos/laborales/competencias-profesionales/son-competencias-profesionales/

Esteban, D. (25 de Agosto de 2022). *Medium*. Obtenido de https://medium.com/@dcortes.net/introducci%C3%B3n-a-nestjs-88f1ca90df35

Etecé, E. (22 de Octubre de 2021). *concepto.de*. Obtenido de https://concepto.de/sistema/

Fernández Carrasco, O. M., García León, D., & Beltrán Benavides, A. (Septiembre de 1995). *Un enfoque actual sobre la calidad del software*. Obtenido de http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1024-94351995000300005

geefe.com. (s.f.). *geefe.com*. Recuperado el 10 de Marzo de 2022, de https://grupoconsultorefe.com/servicio/tecnologias-de-la-informacion/sistemas-web

Gil, C. A. (06 de Mayo de 2012). *EOI*. Obtenido de https://www.eoi.es/blogs/mbaftmad/el-modelo-cocomo-para-estimar-costes-en-un-proyecto-de-software/

Gobierno de Canarias. (s.f.). *¿Qué son las Competencias?* Recuperado el 10 de Marzo de 2023, de https://www.gobiernodecanarias.org/educacion/web/ensenanzas/competencias/que-son-las-competencias/

hostingplus. (14 de Diciembre de 2020). *Qué es MariaDB y cuáles son sus características*. Obtenido de https://www.hostingplus.pe/blog/que-es-mariadb-y-cuales-son-sus-caracteristicas/

IBM. (s.f.). *¿Qué es la prueba de software?* . Obtenido de https://www.ibm.com/es-es/topics/software-testing

IOE Business School. (s.f.). Obtenido de https://www.grupoioe.es/perfil-profesional/

iso25000. (2022). *ISO/IEC 25010*. Obtenido de https://iso25000.com/index.php/normas-iso-25000/iso-25010

isotools. (s.f.). *ISO 27001: Seguridad informática y seguridad de la información*. Obtenido de https://www.isotools.us/2015/01/05/iso-27001-seguridad-informatica-seguridad-informacion/

Jwt.io. (2013). *Introduction to JSON Web Tokens*. Obtenido de https://jwt.io/introduction

Leyva, A. L. (2021). Sistema Multiplataforma para el Acompañamiento y Seguimiento de las competencias en Estudiantes de Instituciones Educativas Básicas Públicas. *Tesis para obtener el Título Profesional de Ingeniero de Sistemas.* Universidad Peruana Union, Tarapoto.

Lucas, J. (4 de Septiembre de 2019). *Qué es NodeJS y para qué sirve*. Obtenido de https://openwebinars.net/blog/que-es-nodejs/

MariaDB Foundation. (2022). *MariaDB Foundation*. Obtenido de https://mariadb.org/es/

Martinez, A. (24 de Mayo de 2022). *Concepto Definición*. Obtenido de https://conceptodefinicion.de/sistema/

MDN. (2 de Diciembre de 2022). *MDN*. Obtenido de https://developer.mozilla.org/es/docs/Glossary/World_Wide_Web

Meléndez Valladarez, S., Gaitan, M., & Pérez Reyes, N. (2016). Metodologia Ágil De Desarrollo De Software Programacion Extrema.

Microsoft. (09 de 02 de 2023). *¿Qué es Git?* Obtenido de https://learn.microsoft.com/es-es/devops/develop/git/what-is-git

Mongodb. (2022). *What is MongoDB?* Obtenido de https://www.mongodb.com/docs/manual/

NestJS . (s.f.). *Introduction*. Obtenido de https://docs.nestjs.com/

NestJS. (2022). *Documentation | NestJS - a Progressive Node.js Framework*. Obtenido de https://docs.nestjs.com/

Normas ISO. (2017). *Normas ISO*. Obtenido de https://www.normas-iso.com/iso-27001/

pirani. (s.f.). Obtenido de https://www.piranirisk.com/es/academia/especiales/iso-27001-que-es-y-como-implementarla

PMOinformatica.com. (27 de Agosto de 2018). *10 Técnicas de estimación de software* . Obtenido de http://www.pmoinformatica.com/2018/08/tecnicas-estimacion-software.html

Redhat. (8 de Mayo de 2020). *¿Qué es una API de REST?* Obtenido de https://www.redhat.com/es/topics/api/what-is-a-rest-api

Ricardo, B. C. (2020). Sistema informático transaccional orientado a la web para el control académico en el instituto labriega mistral de la ciudad de santo domingo. *Proyecto de investigacion previo a la obtención del título de ingeniero en sistemas e informática.* Universidad Regional Autonoma De Los Andes, Santo Domingo - Ecuador.

Rodriguez Poma, H. E. (2021). Sistema de encuesta web para el seguimiento a los titulados de la Universidad Mayor de San Andrés. *Proyecto de Grado para obtener el Título de Licenciatura en Informática.* Universidad Mayor de San Andrés, La paz, Bolivia.

Santander. (20 de Diciembre de 2020). *Metodologías de desarrollo de software: ¿qué son?* Obtenido de https://www.becas-santander.com/es/blog/metodologias-desarrollo-software.html

SINNAPS. (2020). *SINNAPS*. Obtenido de https://www.sinnaps.com/blog-gestion-proyectos/metodologia-xp#fase-1-planificacion

Sotomayor, S. G. (9 de Diciembre de 2021). *IEBS*. Obtenido de Las metodologías ágiles más utilizadas y sus ventajas dentro de la empresa: https://www.iebschool.com/blog/que-son-metodologias-agiles-agile-scrum/

Techopedia. (2 de Septiembre de 2022). *Techopedia*. Obtenido de https://www.techopedia.com/definition/5613/web

Thales. (s.f.). *¿Qué es la seguridad del software y por qué es tan importante ahora?* Obtenido de https://cpl.thalesgroup.com/es/software-monetization/what-is-software-security

ThemeSphere. (15 de Diciembre de 2021). *¿Qué es TypeScript y qué relación tiene con JavaScript?* Obtenido de https://blog.talenthackers.net/que-es-typescript-y-que-relacion-tiene-con-javascript

Tilio, A. (s.f.). *De Significados*. Obtenido de https://designificados.com/sistema/

Tola Mendoza, M. (2020). “Sistema de información web para el control y seguimiento académico” caso: Instituto técnico San Pablo. *Para Optar al Título de Licenciatura en Ingeniería de Sistemas.* Universidad Pública del Alto, El Alto, Bolivia.

Universidad Publica De El Alto. (s.f.). *Estatuto Organico Reglamentos Generales Resoluciones del II Congreso.*

UPEA Carrera de Medicina – Universidad Pública de El Alto. (2022). Obtenido de https://www.medicinaupea.com/

Veryti. (28 de Julio de 2022). *La ISO/IEC 9126: 2001: Características de la calidad de software*. Obtenido de https://www.verity.cl/que-es-norma-iso-iec-9126-2001/

Villarroel Apaza, V. (2020). Sistema web para la gestión de procesos de pasantias y practicas profesionales. *Para optar al título de Licenciatura en Ingeniería de Sistemas.* Universidad Publica de el Alto, El Alto, Bolivia.

ANEXOS

ANEXO A

AVALES

El Alto, 24 de noviembre de 2023

Señor:
ING. WILLIAM ROQUE ROQUE
DIRECTOR DE CARRERA
INGENIERÍA DE SISTEMAS
Presente. -

REF. AVAL DE CONFORMIDAD

Distinguido directo de carrera:

Mediante la presente tengo a bien de comunicarle mi conformidad del Trabajo de Grado:

TÍTULO: SISTEMA WEB DE SEGUIMIENTO INDIVIDUAL UNIVERSITARIO PARA LA EVALUACIÓN DE COMPETENCIAS ADQUIRIDAS TAL QUE VERIFIQUE EL PERFIL PROFESIONAL
CASO: CARRERA DE MEDICINA UNIVERSIDAD PÚBLICA DE EL ALTO
MODALIDAD: PROYECTO DE GRADO
UNIV. OTMAR KEVIN AGUILAR LEON
Registro universitario: 200007299
Cedula de identidad: 9991405

Para su defensa publica y evaluación correspondiente a la materia de Taller de Grado II, de acuerdo al reglamento vigente de la carrera de Ingeniería de Sistemas de la Universidad Pública de El Alto.

Atentamente,

ING. Dionicio Henry Pacheco Rios
TUTOR METODOLÓGICO
TALLER DE GRADO II

El Alto, 24 de noviembre de 2023

Señor:
ING. DIONICIO HENRY PACHECO RIOS
TUTOR METODOLÓGICO
TALLER DE GRADO II
Presente. -

REF. AVAL DE CONFORMIDAD

Distinguido tutor metodológico:

Mediante la presente tengo a bien de comunicarle mi conformidad del Trabajo de Grado:

TÍTULO: SISTEMA WEB DE SEGUIMIENTO INDIVIDUAL UNIVERSITARIO PARA LA EVALUACIÓN DE COMPETENCIAS ADQUIRIDAS TAL QUE VERIFIQUE EL PERFIL PROFESIONAL
CASO: CARRERA DE MEDICINA UNIVERSIDAD PÚBLICA DE EL ALTO
MODALIDAD: PROYECTO DE GRADO
UNIV. OTMAR KEVIN AGUILAR LEON
Registro universitario: 200007299
Cedula de identidad: 9991405

Para su defensa publica y evaluación correspondiente a la materia de Taller de Grado II, de acuerdo al reglamento vigente de la carrera de Ingeniería de Sistemas de la Universidad Pública de El Alto.

Atentamente,

.....
Lic. Roger Navia Gutierrez
TUTOR ESPECIALISTA

El Alto, 23 de noviembre de 2023

Señor:
ING. DIONICIO HENRY PACHECO RIOS
TUTOR METODOLÓGICO
TALLER DE GRADO II
Presente. -

REF. AVAL DE CONFORMIDAD

Distinguido tutor metodológico:

Mediante la presente tengo a bien de comunicarle mi conformidad del Trabajo de Grado:

TÍTULO: SISTEMA WEB DE SEGUIMIENTO INDIVIDUAL UNIVERSITARIO PARA LA EVALUACIÓN DE COMPETENCIAS ADQUIRIDAS TAL QUE VERIFIQUE EL PERFIL PROFESIONAL
CASO: CARRERA DE MEDICINA UNIVERSIDAD PÚBLICA DE EL ALTO
MODALIDAD: PROYECTO DE GRADO
UNIV. OTMAR KEVIN AGUILAR LEON
Registro universitario: 200007299
Cedula de identidad: 9991405

Para su defensa pública y evaluación correspondiente a la materia de Taller de Grado II, de acuerdo al reglamento vigente de la carrera de Ingeniería de Sistemas de la Universidad Pública de El Alto.

Atentamente,

.....
M.Sc. Lic. Wendy Yomar Sarmiento Martínez
TUTOR REVISOR

Universidad Pública de El Alto

Creada por Ley 2115 del 5 de Septiembre de 2000 y Autónoma por Ley 2556 del 12 de Noviembre de 2003

CITE: UPEA/CMED/NOTAS INTERNAS N° 298/2023
El Alto, 08 de noviembre de 2023

Señor:
Ing. Dionicio Henry Pacheco Ríos
TUTOR METODOLÓGICO – TALLER DE GRADO II
UNIVERSIDAD PÚBLICA DE EL ALTO
Presente. -

Ref.: **AVAL DE CONFORMIDAD**

Distinguido tutor metodológico:

Mediante la presente tengo a bien felicitar por el buen trabajo realizado en su asignatura, y comunicarle mi conformidad con el Trabajo de Grado Titulado: *SISTEMA WEB DE SEGUIMIENTO INDIVIDUAL UNIVERSITARIO PARA LA EVALUACIÓN DE COMPETENCIAS ADQUIRIDAS TAL QUE VERIFIQUE EL PERFIL PROFESIONAL.*

Caso: CARRERA DE MEDICINA UNIVERSIDAD PÚBLICA DE EL ALTO
Modalidad: PROYECTO DE GRADO
Univ.: OTMAR KEVIN AGUILAR LEON
Registro Universitario: 200007299
Cedula de Identidad: 9991405 LP.

De tal forma cabe recalcar que el sistema satisface los requerimientos de la Carrera de Medicina, de esta forma se dio cumplimiento de los objetivos del presente trabajo e implementado satisfactoriamente se realizó las capacitaciones necesarias en la Institución.

Es cuanto certifico en honor a la verdad para fines consiguientes del interesado para su defensa publica y evaluación correspondiente a la materia de Taller de Grado II de acuerdo al reglamento vigente de la Carrera de Ingeniería de Sistemas de la Universidad Pública de El Alto.

Atentamente,

Dr. Jhonny Santos Rojas Roque
DIRECTOR
CARRERA MEDICINA
UNIVERSIDAD PÚBLICA DE EL ALTO

UNIVERSIDAD PÚBLICA DE EL ALTO
CARRERA DE MEDICINA
DIRECCION
ACREDITADA
El Alto - Bolivia

Dir.: Av. Sucre A s/n Villa Esperanza Telf.: (591-2) 2-844177 - Fax.:(591-2) 2-845800 www.upea.edu.bo

ANEXO B

COMPETENCIAS DE LA CARRERA DE MEDICINA

Competencias de la carrera de medicina

Macro competencias del médico general egresado de la UPEA:

1 Realiza la atención integral en salud al individuo, familia y comunidad.
2 Desarrolla una interrelación efectiva con el paciente, la familia y la comunidad.
3 Utiliza la evidencia científica en la práctica profesional.
4 Incorpora la interculturalidad en su actividad médica
5 Incorpora el enfoque de calidad en su práctica profesional
6 Aplica los principios éticos y legales en la práctica de la medicina.
7 Trabaja en equipo en los sistemas de salud.
8 Entiende el proceso salud enfermedad desde sus determinantes
9 Realiza investigación en Salud
10 Participa activamente en las políticas, planes y programas de Salud.

Competencias genéricas Turing:

Todos los profesionales deben tener las siguientes capacidades

1 Capacidad de abstracción, análisis y síntesis.
2 Capacidad de aplicar los conocimientos en la práctica.
3 Capacidad para organizar y planificar el tiempo.
4 Conocimientos sobre el área de estudio y la profesión.
5 Responsabilidad social y compromiso ciudadano.
6 Capacidad de comunicación oral y escrita.
7 Capacidad de comunicación en un segundo idioma.
8 Habilidades en el uso de las tecnologías de la información y de la comunicación.
9 Capacidad de investigación.
10 Capacidad de aprender y actualizarse permanentemente
11 Habilidades para buscar, procesar y analizar información procedente de fuentes diversas.
12 Capacidad crítica y autocrítica.
13 Capacidad para actuar en nuevas situaciones.
14 Capacidad creativa.
15 Capacidad para identificar, plantear y resolver problemas.
16 Capacidad para tomar decisiones.
17 Capacidad de trabajo en equipo.
18 Habilidades interpersonales.
19 Capacidad de motivar y conducir hacia metas comunes.
20 Compromiso con la preservación del medio ambiente.
21 Compromiso con su medio socio-cultural.
22 Valoración y respeto por la diversidad y multiculturalidad.
23 Habilidad para trabajar en contextos internacionales.
24 Habilidad para trabajar en forma autónoma.
25 Capacidad para formular y gestionar proyectos.
26 Compromiso ético.

27 Compromiso con la calidad.

Competencias específicas de medicina:

Al finalizar los estudios de Medicina los egresados deben tener la capacidad de:

1 Capacidad para llevar a cabo la práctica clínica
2 Capacidad para proveer atención medica de urgencias
3 Capacidad para prescribir medicamentos
4 Capacidad para comunicarse en su ejercicio profesional
5 Capacidad para realizar procedimientos diagnósticos y terapéuticos
6 Capacidad para identificar los factores determinantes en el proceso de salud-enfermedad
7 Capacidad para el uso de la evidencia en la práctica médica
8 Capacidad para el uso de la información y sus tecnologías efectivamente en un contexto médico
9 Capacidad para aplicar los principios éticos y legales en la práctica de la medicina
10 Capacidad para trabajar efectivamente en los sistemas de salud

Desarrollo de las competencias específicas de medicina:

Al finalizar los estudios de Medicina los egresados deben tener la capacidad de

Capacidad para llevar a cabo la práctica clínica

1. Capacidad para redactar la Historia Clínica.
2. Capacidad para realizar en cualquier ambiente la anamnesis completa, enfatizando los aspectos psicosociales y ambientales que inciden en la salud de las personas.
3. Capacidad para realizar el examen físico completo incluyendo la evaluación del estado mental.
4. Capacidad para realizar el diagnóstico sindromático y formular hipótesis diagnósticas teniendo en cuenta, los datos anamnésicos, los hallazgos del examen físico y las enfermedades prevalentes.
5. Capacidad para plantear diagnósticos diferenciales.
6. Capacidad para seleccionar, indicar e interpretar las pruebas diagnósticas.
7. Capacidad para indicar y realizar los tratamientos médicos correspondientes.
8. Capacidad para derivar a otro nivel de atención.

Capacidad para proveer atención medica de urgencias

9. Capacidad para reconocer, evaluar y categorizar las emergencias médicas.
10. Capacidad para manejar la fase inicial de la emergencia médica.
11. Capacidad para proveer primeros auxilios.
12. Capacidad para proveer soporte vital básico y reanimación cardio cerebro pulmonar.
13. Capacidad para proveer soporte vital avanzado.
14. Capacidad para proveer cuidado al paciente con trauma.

Capacidad para prescribir medicamentos

15. Capacidad para seleccionar los medicamentos indicados según el contexto clínico.
16. Capacidad para prescribir de manera clara, precisa y segura.
17. Capacidad para reconocer y manejar los eventos adversos.

Capacidad para comunicarse en su ejercicio profesional

18. Capacidad para comunicarse de manera eficaz oralmente, por escrito y en forma no verbal teniendo en cuenta la diversidad y las limitaciones que pueden dificultar la comunicación con:
    • los pacientes
    • la familia
    • el equipo de salud
    • la comunidad
19. Capacidad para comunicar la naturaleza y severidad del padecimiento
20. Capacidad para obtener el consentimiento informado cuando corresponda

Capacidad para realizar procedimientos diagnósticos y terapéuticos

21. Capacidad para evaluar signos vitales.
22. Capacidad para realizar venopunción.
23. Capacidad para realizar canalización venosa.
24. Capacidad para administrar medicamentos por las diferentes vías.
25. Capacidad para realizar intubación orotraqueal y soporte vital básico.
26. Capacidad para colocar sondas.
27. Capacidad para realizar cuidados de ostomías.
28. Capacidad para realizar punción supra púbica.
29. Capacidad para realizar toracentesis, paracentesis y punción lumbar.
30. Capacidad para realizar un electrocardiograma.
31. . Capacidad para atender un parto eutócico.
32. Capacidad para realizar especuloscopia, tacto vaginal y toma de citología.
33. Capacidad para realizar tacto rectal.
34. Capacidad para realizar taponamiento nasal anterior.
35. Capacidad para realizar maniobras hemostáticas iniciales ante hemorragia externa.
36. Capacidad para realizar suturas, curaciones de heridas y drenaje de abscesos.
37. Capacidad para mover, inmovilizar y transportar pacientes.

Capacidad para identificar los factores determinantes en el proceso de salud-enfermedad

38. Capacidad para identificar los factores psicológicos (stress, dependencia y abuso de alcohol, drogas y tabaco).
39. Capacidad para identificar los factores sociales (violencia, accidentes, maltrato, abuso, marginación, discriminación).
40. Capacidad para identificar los factores económicos (pobreza, inequidad).

41. Capacidad para identificar los factores ambientales (contaminación, clima, destrucción del ecosistema).

Capacidad para el uso de la evidencia en la práctica médica

42. Capacidad para analizar críticamente la literatura científica.
43. Capacidad para aplicar el análisis estadístico de los datos.
44. Capacidad para realizar medicina basada en la evidencia

Capacidad para el uso de la información y sus tecnologías efectivamente en un contexto médico

45. Capacidad para el uso de computadores.
46. Capacidad para acceder a las fuentes de información.
47. Capacidad para guardar en forma completa y segura los registros médicos.

Capacidad para aplicar los principios éticos y legales en la práctica de la medicina

48. Capacidad para aplicar principios y análisis éticos en el ejercicio clínico.
49. Capacidad para obtener y registrar el consentimiento informado.
50. Capacidad para mantener la confidencialidad.
51. Capacidad de respeto a la diversidad.
52. Capacidad para respetar los derechos del paciente, del equipo de salud y de la comunidad.
53. Capacidad para respetar y brindar cuidados al paciente terminal.
54. Capacidad para expedir certificados de acuerdo con la legislación.
55. Capacidad para informar las enfermedades de notificación obligatoria.

Capacidad para trabajar efectivamente en los sistemas de salud

56. Capacidad para reconocer la estructura y funcionamiento del sistema de salud.
57. Capacidad para administrar y gestionar los distintos sistemas de salud de la población.
58. Capacidad para participar efectiva y activamente dentro del equipo de salud y en la comunidad.
59. Capacidad para reconocer y aplicar las políticas y programas de salud del país.
60. Capacidad para reconocer y gestionar los recursos para la atención en salud.
61. Capacidad para reconocer el perfil epidemiológico de la población.
62. Capacidad para reconocer y aplicar los principios de promoción de la salud y prevención de enfermedades
63. Capacidad para conocer, aplicar, y respetar las normas de bioseguridad.

ANEXO C

PRUEBAS DE ACEPTACIÓN

Tarjeta de Prueba de aceptación N° 2

N° Prueba de Aceptación: 2
N° Historia de usuario: 2
Descripción: El administrador crea un nuevo estudiante en el sistema

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos personales del estudiante
- Solo el administrador puede crear estudiantes

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de estudiante
3. Pulsa el botón nuevo en el módulo de estudiante
4. Introduce los datos del estudiante
5. Si el estudiante no está registrado se guardará en el sistema caso contrario saldrá una alerta con el mensaje "El usuario ya existe"
6. El estudiante aparece en la lista de estudiantes

Evaluación: Aprobada
Aprobada por:

Dr. Reynaldo Tonconi Montoya
Docente Medicina

Pruebas de Aceptación

Las siguientes pruebas fueron realizadas conjuntamente con los encargados del seguimiento en el desarrollo del sistema.

Tarjeta de Prueba de aceptación N° 1

N° Prueba de Aceptación: 1
N° Historia de usuario: 1
Descripción: El usuario accede al sistema a la pantalla de inicio de sesión e intenta acceder con sus credenciales de acceso.
Condiciones de ejecución:

- El usuario debe estar dado de alta por el administrador.
- El usuario debe contar con sus credenciales de acceso.

Pasos para de ejecución:

1. El usuario ingresa al sistema
2. El usuario llena los campos nombre de usuario y contraseña
3. Si las credenciales son correctas el sistema redirecciona a la página de inicio del sistema caso contrario no se ingresa vuelve a intentar el paso 2

Evaluación: Aprobada
Aprobada por:

Dr. Reynaldo Tonconi Montoya
Docente Medicina

Tarjeta de Prueba de aceptación N° 4

N° Prueba de Aceptación: 4

N° Historia de usuario: 2

Descripción: El administrador elimina uno o varios estudiantes

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos personales del estudiante
- Solo el administrador puede eliminar estudiantes
- El estudiante debe está registrado previamente

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de estudiante
3. Busca y selecciona al estudiante/es que va a eliminar
4. Pusa el botón eliminar en la sección de acciones
5. Pulsa el botón eliminar
6. Se guardan los cambios en los datos

Evaluación: Aprobada

Aprobada por:

Tarjeta de Prueba de aceptación N° 3

N° Prueba de Aceptación: 3

N° Historia de usuario: 2

Descripción: El administrador modifica datos de un estudiante

Condiciones de ejecución:

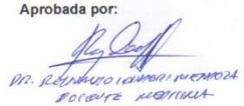
- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos personales del estudiante
- Solo el administrador puede editar estudiantes
- El estudiante debe esta registrado previamente

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de estudiante
3. Busca al estudiante que requiere una modificación de sus datos
4. Pusa el botón modificar en la sección de acciones
5. Introduce los datos del estudiante
6. Pulsa el botón guardar
7. Se guardan los cambios en los datos

Evaluación: Aprobada

Aprobada por:



Tarjeta de Prueba de aceptación N° 6

N° Prueba de Aceptación: 6

N° Historia de usuario: 3

Descripción: El administrador modifica datos de un docente

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos personales del estudiante
- Solo el administrador puede editar docentes
- El docente debe esta registrado previamente

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de docentes
3. Busca al docente que requiere una modificación de sus datos
4. Pusa el botón modificar en la sección de acciones
5. Introduce los datos del docente
6. Pulsa el botón guardar
7. Se guardan los cambios en los datos

Evaluación: Aprobada

Aprobada por:

Tarjeta de Prueba de aceptación N° 5

N° Prueba de Aceptación: 5

N° Historia de usuario: 3

Descripción: El administrador crea un nuevo docente en el sistema

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos personales del estudiante
- Solo el administrador puede crear docentes

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de docentes
3. Pulsa el botón nuevo en el módulo de docentes
4. Introduce los datos del docente
5. Si el docente no está registrado se guardará en el sistema caso contrario saldrá una alerta con el mensaje "El usuario ya existe"
6. El nuevo docente aparece en la lista de docentes

Evaluación: Aprobada

Aprobada por:

Tarjeta de Prueba de aceptación N° 8

N° Prueba de Aceptación: 8

N° Historia de usuario: 4

Descripción: El administrador crea nueva competencia

Condiciones de ejecución:

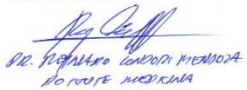
- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos de la competencia

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de competencias
3. Pulsa en el botón nuevo en el módulo de competencias
4. Introduce los datos de la competencia
5. Pulsa en guardar
6. La nueva competencia se lista en las competencias

Evaluación: Aprobada

Aprobada por:



Tarjeta de Prueba de aceptación N° 7

N° Prueba de Aceptación: 7

N° Historia de usuario: 3

Descripción: El administrador elimina uno o varios docentes

Condiciones de ejecución:

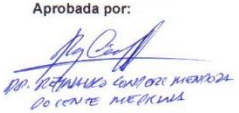
- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos personales del docente
- Solo el administrador puede eliminar estudiantes
- El docente debe esta registrado previamente

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de docente
3. Busca y selecciona al docente/es que va a eliminar
4. Pusa el botón eliminar en la sección de acciones
5. Pulsa el botón eliminar
6. Se guardan los cambios en los datos

Evaluación: Aprobada

Aprobada por:



Tarjeta de Prueba de aceptación N° 10

N° Prueba de Aceptación: 10

N° Historia de usuario: 4

Descripción: El administrador elimina una o varias competencias

Condiciones de ejecución:

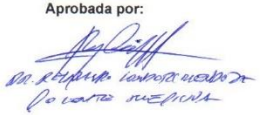
- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos de la competencia que requiere eliminar

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de competencias
3. Busca y selecciona la competencia que requiere eliminar en los datos
4. Pulsa en el botón eliminar en el en apartado de acciones en el módulo de competencias
5. Se elimina la competencia

Evaluación: Aprobada

Aprobada por:



Tarjeta de Prueba de aceptación N° 9

N° Prueba de Aceptación: 9

N° Historia de usuario: 4

Descripción: El administrador modifica una competencia

Condiciones de ejecución:

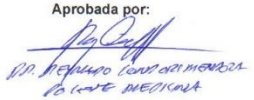
- El administrador debe estar con una sesión iniciada
- El administrador debe tener los nuevos datos de la competencia

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de competencias
3. Busca la competencia que requiere modificación en los datos
4. Pulsa en el botón editar en el en apartado de acciones en el módulo de competencias
5. Introduce los datos de la competencia
6. Pulsa en guardar
7. La nueva competencia se lista en las competencias

Evaluación: Aprobada

Aprobada por:



Tarjeta de Prueba de aceptación N° 12

N° Prueba de Aceptación: 12

N° Historia de usuario: 5

Descripción: El administrador modifica una asignatura

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los nuevos datos de la asignatura

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de asignaturas
3. Busca la competencia que requiere modificación en los datos
4. Pulsa en el botón editar en el en apartado de acciones en el módulo de asignaturas
5. Introduce los datos de la asignatura
6. Pulsa en guardar
7. La nueva asignatura se lista en las asignaturas

Evaluación: Aprobada

Aprobada por:

Dr. Reynaldo Contreras Mendoza
Docente Medicina

Tarjeta de Prueba de aceptación N° 11

N° Prueba de Aceptación: 11

N° Historia de usuario: 5

Descripción: El administrador crea nueva asignatura

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos de la asignatura

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de asignaturas
3. Pulsa en el botón nuevo en el módulo de asignaturas
4. Introduce los datos de la asignatura
5. Pulsa en guardar
6. La nueva asignatura se lista en las asignaturas

Evaluación: Aprobada

Aprobada por:

Dr. Reynaldo Contreras Mendoza
Docente Medicina

Tarjeta de Prueba de aceptación N° 14

N° Prueba de Aceptación: 14

N° Historia de usuario: 5

Descripción: El administrador asigna competencia a una asignatura

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos de la asignatura
- Las competencias deben estar previamente creadas

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de asignaturas
3. Busca y selecciona la asignatura que requiere asignar las competencias
4. Pulsa en el botón asignar competencias en el en apartado de acciones en el módulo de asignaturas
5. Selecciona las competencias que desean designar a la asignatura
6. Pulsa el botón guardar

Evaluación: Aprobada

Aprobada por:

Dr. Reynaldo Cenobi Mendoza
Docente Medicina

Tarjeta de Prueba de aceptación N° 13

N° Prueba de Aceptación: 13

N° Historia de usuario: 5

Descripción: El administrador elimina una o varias asignaturas

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos de la asignatura que requiere eliminar

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de asignaturas
3. Busca y selecciona la asignatura que requiere eliminar en los datos
4. Pulsa en el botón eliminar en el en apartado de acciones en el módulo de asignaturas
5. Se elimina la asignatura

Evaluación: Aprobada

Aprobada por:

Dr. Reynaldo Cenobi Mendoza
Docente Medicina

Tarjeta de Prueba de aceptación N° 16

N° Prueba de Aceptación: 16

N° Historia de usuario: 7

Descripción: El docente crea una calificación de parcial en una asignatura

Condiciones de ejecución:

- El docente debe estar con una sesión iniciada
- El docente debe tener por lo menos una asignatura
- La asignatura debe tener inscritos a los estudiantes

Pasos para de ejecución:

1. Ingresa a las asignaturas
2. Selecciona la asignatura donde se desea crear la calificacion parcial
3. Pulsa el botón crear parcial
4. Introduce la descripción y la calificacion del parcial

Evaluación: Aprobada

Aprobada por:

Dr. Reynaldo Condori Huenaqa
Docente Medicina

Tarjeta de Prueba de aceptación N° 15

N° Prueba de Aceptación: 15

N° Historia de usuario: 6

Descripción: El administrador Elimina calificación

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- El administrador debe tener los datos de la asignatura
- El administrador debe tener datos de la calificación eliminar
- Solo se puede eliminar la calificacion de la asignatura si y solo si ningún estudiante fue calificado en esa instancia de la calificacion

Pasos para de ejecución:

1. Ingresa al panel de administración
2. Ingresa al módulo de asignaturas
3. Busca y selecciona la asignatura donde requiere eliminar la calificacion
4. Selecciona la calificacion que desea eliminar
5. Se elimina la calificacion

Evaluación: Aprobada

Aprobada por:

Dr. Reynaldo Condori Huenaqa
Docente Medicina

Tarjeta de Prueba de aceptación N° 18

N° Prueba de Aceptación: 18

N° Historia de usuario: 7,8

Descripción: El docente califica el parcial o practica de un estudiante (tanto como para parcial y practicas es el mismo procedimiento)

Condiciones de ejecución:

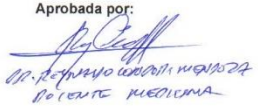
- El docente debe estar con una sesión iniciada
- El docente debe tener por lo menos una asignatura
- La asignatura debe tener inscritos a los estudiantes
- La calificacion ya debe estar creada

Pasos para de ejecución:

1. Ingresa a las asignaturas
2. Selecciona la asignatura donde se desea crear la calificacion parcial
3. Selecciona en botón calificar en la lista de los estudiantes
4. Selecciona la calificacion que quiere calificar
5. Introduce la calificacion obtenida por el estudiante

Evaluación: Aprobada

Aprobada por:



Tarjeta de Prueba de aceptación N° 17

N° Prueba de Aceptación: 17

N° Historia de usuario: 8

Descripción: El docente crea una calificación de practica en una asignatura

Condiciones de ejecución:

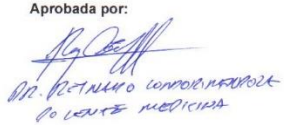
- El docente debe estar con una sesión iniciada
- El docente debe tener por lo menos una asignatura
- La asignatura debe tener inscritos a los estudiantes

Pasos para de ejecución:

5. Ingresa a las asignaturas
6. Selecciona la asignatura donde se desea crear la calificacion practica
7. Pulsa el botón crear parcial
8. Introduce la descripción y la calificacion del practica

Evaluación: Aprobada

Aprobada por:



Tarjeta de Prueba de aceptación N° 20

N° Prueba de Aceptación: 20

N° Historia de usuario: 10

Descripción: El estudiante puede observar las sus calificaciones y competencias obtenidas durante la gestión en curso

Condiciones de ejecución:

- El estudiante debe estar con una sesión iniciada
- El estudiante debe tener por lo menos una asignatura inscrita

Pasos para de ejecución:

1. Ingresa a las calificaciones
2. Se despliega la lista de las calificaciones obtenidas
3. Ingresa a competencias
4. Se despliega la lista de las competencias adquiridas

Evaluación: Aprobada

Aprobada por:

M.D. REYNALDO CONTRERAS MENDOZA
DOCENTE MEDICINA

Tarjeta de Prueba de aceptación N° 19

N° Prueba de Aceptación: 19

N° Historia de usuario: 9

Descripción: El docente adjudica la competencia al estudiante dependiendo de las particularidades de la asignatura.

Condiciones de ejecución:

- El docente debe estar con una sesión iniciada
- El docente debe tener por lo menos una asignatura
- La asignatura debe tener inscritos a los estudiantes
- Las competencias deben están designadas a la asignatura

Pasos para de ejecución:

1. Ingresa a las asignaturas
2. Selecciona la asignatura donde se desea crear la calificacion parcial
3. Pulsa el botón Dar competencia en la lista de los estudiantes
4. Selecciona las una o varias competencias las cuales se adjudicarán al estudiante
5. Pulsa el botón guarda

Evaluación: Aprobada

Aprobada por:

M.D. REYNALDO CONTRERAS MENDOZA
DOCENTE MEDICINA

Tarjeta de Prueba de aceptación N° 22

N° Prueba de Aceptación: 22

N° Historia de usuario: 12

Descripción: El administrador inscribe a un estudiante a las asignaturas

Condiciones de ejecución:

- El administrador debe estar con una sesión iniciada
- Las asignaturas deben estar ya creadas
- La cuanta de estudiante ya debe estar creada

Pasos para de ejecución:

1. Ingresa a inscripciones
2. Pulsa el botón inscribir
3. Selecciona al estudiante que desea inscribir
4. Selecciona uno a varias asignaturas a las que desea inscribir al estudiante del paso 4
5. Pulsa Guardar

Evaluación: Aprobada

Aprobada por:

Dr. Reinaldo Condori Mendoza
Docente Medicina

Tarjeta de Prueba de aceptación N° 21

N° Prueba de Aceptación: 21

N° Historia de usuario: 11

Descripción: El usuario cierra la sesión

Condiciones de ejecución:

- El usuario debe estar con una sesión iniciada

Pasos para de ejecución:

1. Pulsa el botón salir
2. Es redireccionado al módulo de inicio de sesión

Evaluación: Aprobada

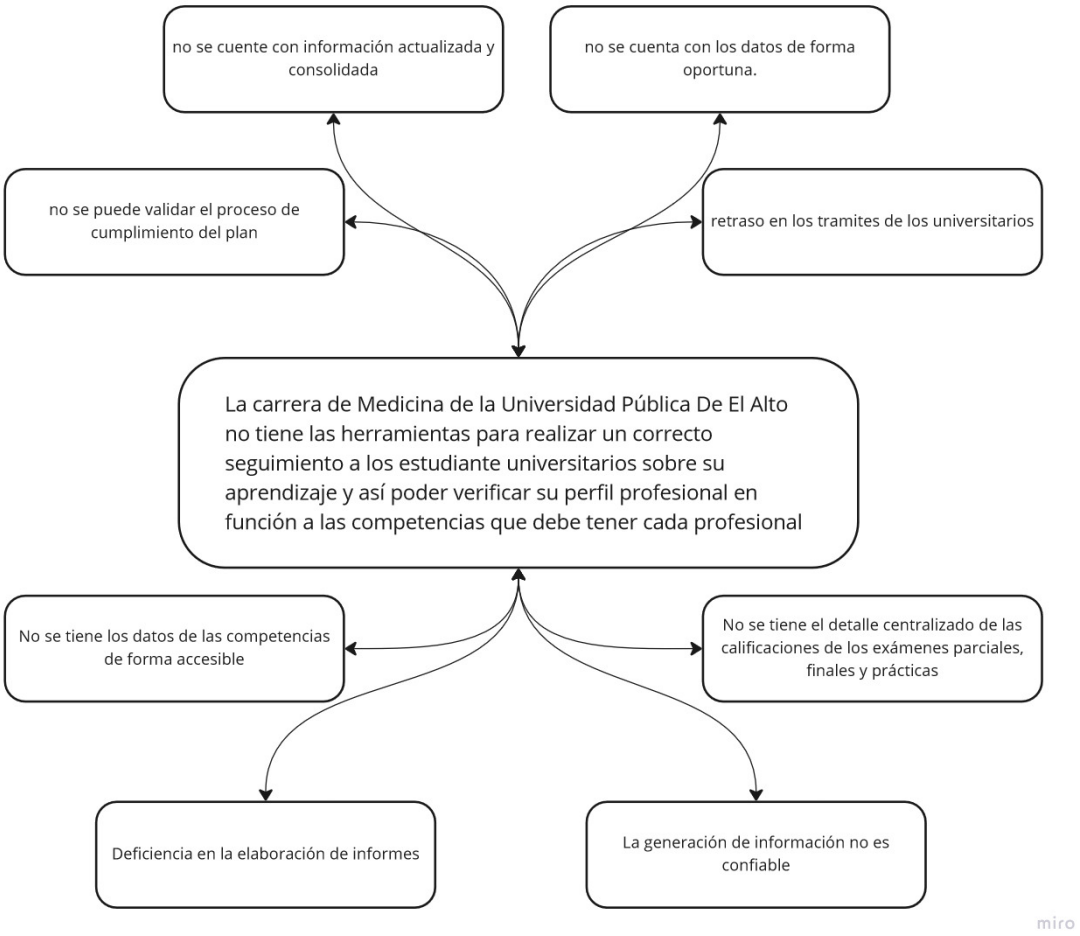
Aprobada por:

M.C. Reinaldo Condori Mendoza
Docente Medicina

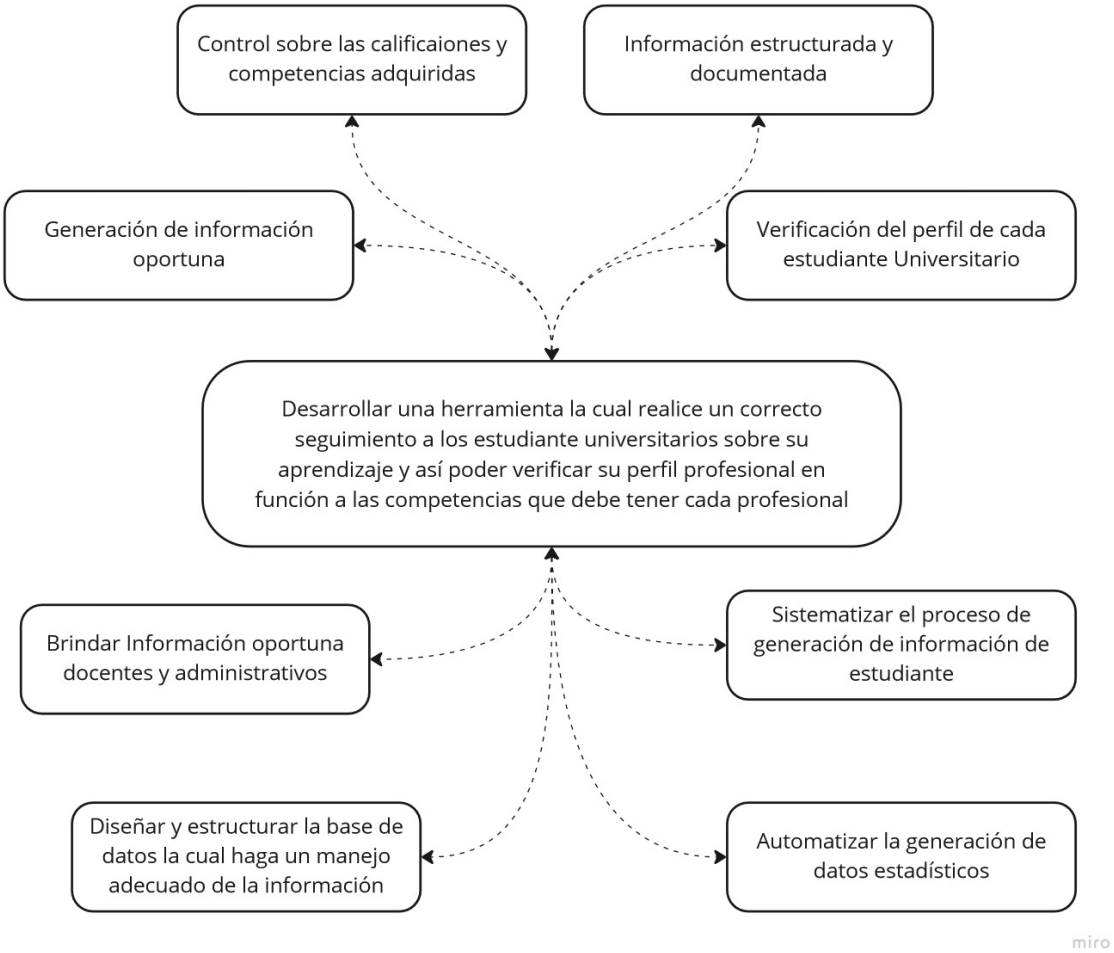
ANEXO D

ÁRBOL DE PROBLEMAS Y OBJETIVOS

Arbol de Problemas



Arbol de Objetivos



ANEXO E

MANUALES

SISEIN

MANUAL DE USUARIO
ESTUDIANTE

SISTEMA DE CONTROL INDIVIDUAL ESTUDIANTIL PARA LA VERIFICACIÓN DE PERFIL PROFESIONAL

Versión 1.0

GUÍA RAPADA DE USO DEL SISTEMA DE CONTROL INDIVIDUAL ESTUDIANTIL PARA LA VERIFICACIÓN DEL PERFIL

¿Qué es el sistema SISEIN?

El sistema SISEIN que implementado en el la carrera de medicina de la universidad publica de el Alto, Nos permite llevar el control del perfil profesional que debe cumplir cada estudiante, verificando e informando la calificaciones y competencias d cada estudiante

Los docentes tienen el control sobre la asignatura la cual desarrollan sus clases ya se creando sus calificaciones como ser practicas y parciales a si mismo pueden adjudicar a los estudiantes las competencias correspondientes a la Asignatura.

Los estudiantes pueden ver sus calificaciones y competencias adquiridas de manera inmediata en el sistema

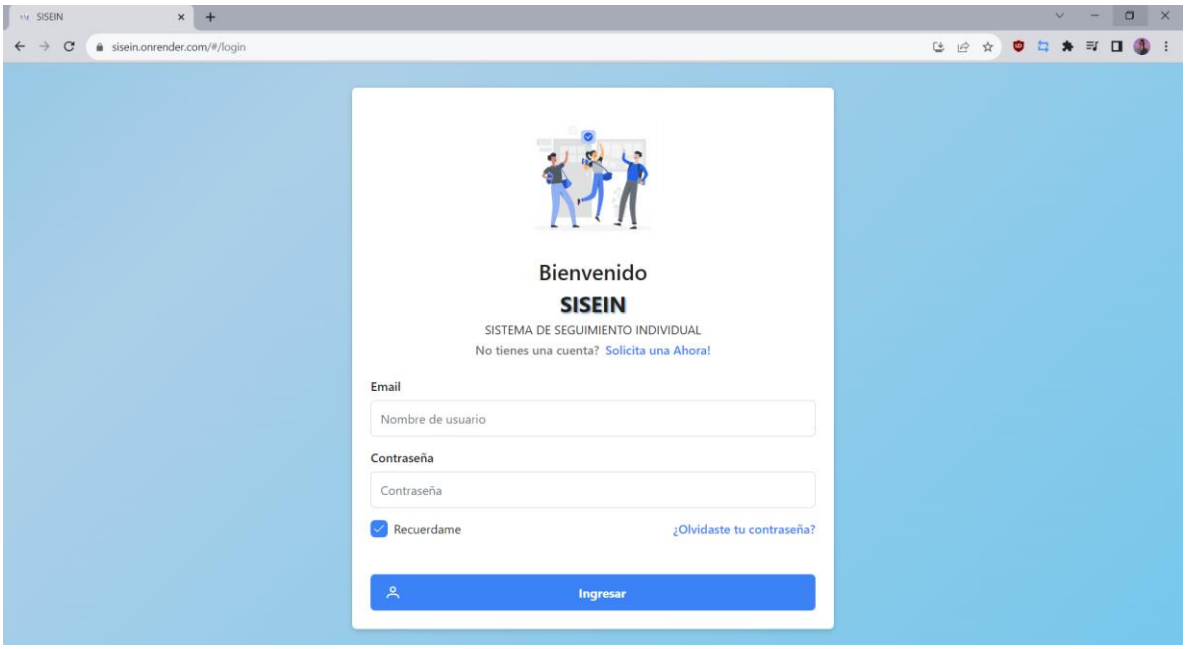
¿Cómo puedo acceder al sistema?

Se puede acceder al sistema desde cualquier navegador de internet (Microsoft Edge, Firefox, Chrome,etc) desde cualquier dispositivo (Computadora, Smartphone ,Tablet) que tenga acceso a internet.

Se puede acceder des la URL:

https://sisein.onrender.com/#/login

Al ingresar al sitio nos redireccionar a la pagina de inicio de sesion



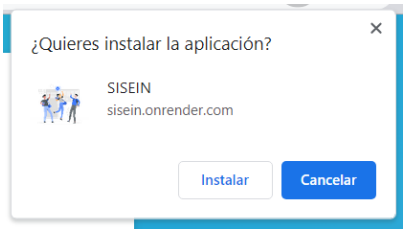
Instalación del sistema (Opcional):

Al momento de ingresar al sistema desde un navegador Chrome nos aparecerá la opción de instalar el sistema en nuestro dispositivo, para ello debemos realizar los siguientes pasos:

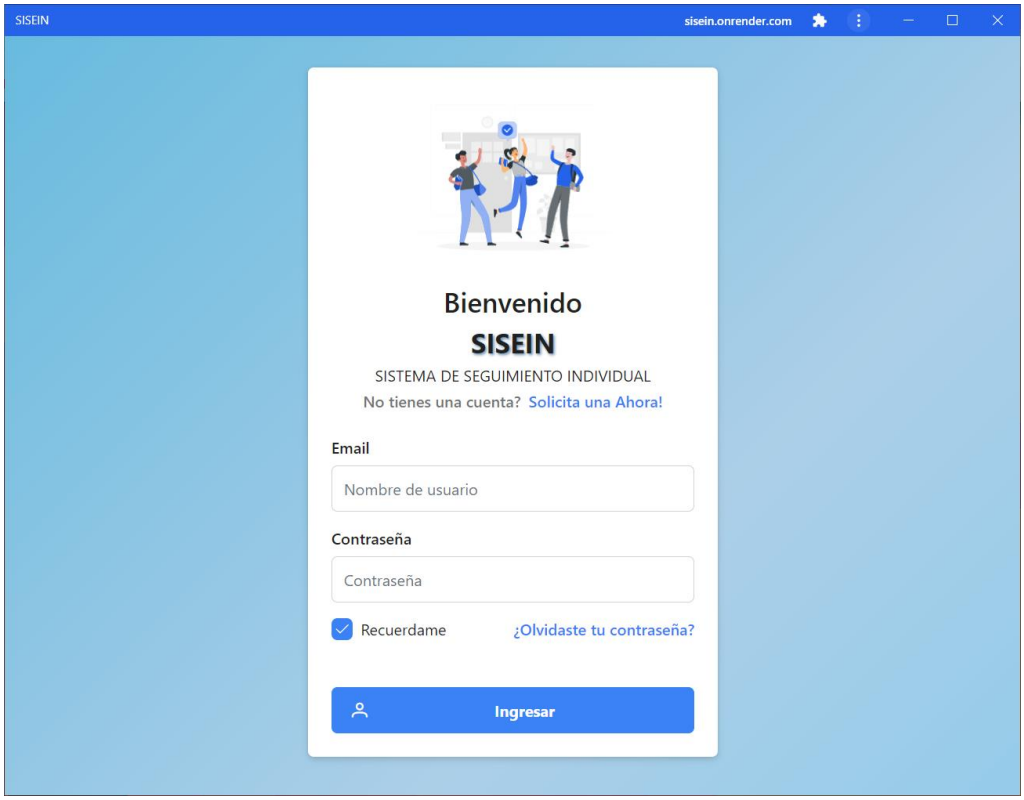
Instalar en Windows

1. Click en boton

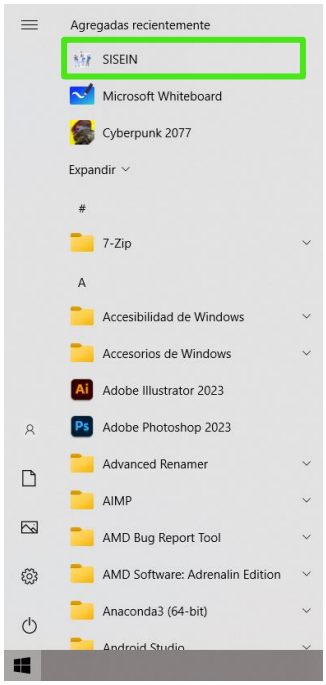
2. Click en Instalar

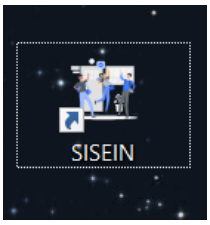


3. Aparece la siguiente ventana



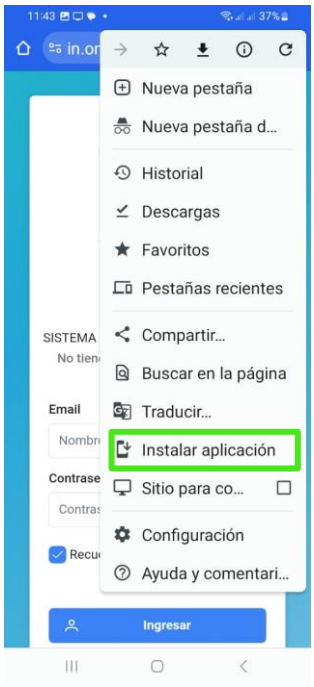
La aplicación se encontrará en tu escritorio y también en tu menú de aplicaciones



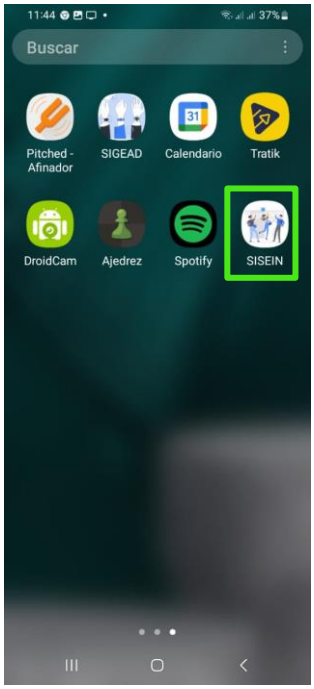


Instalación Android

1. Ingresamos al sistema y hacemos click en el botón mostrar más (tres puntos) del navegados y le damos click en instalar aplicación



2. Click en instalar la esperamos un momento y en el menú de aplicaciones aparece la aplicación instalada



Inicio de sesión

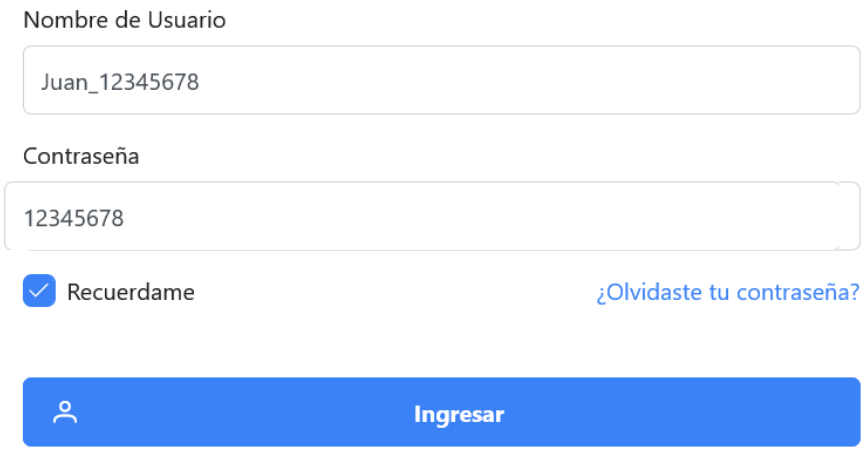
Para iniciar sesión se requiere las credenciales de acceso los cuales son:

Nombre de usuario: PrimerNombre_Carnet de identidad
Contraseña: Carnet de Identidad

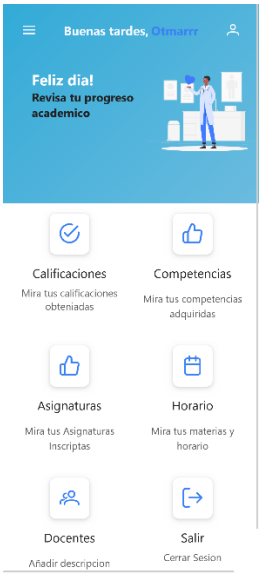
Ejemplo:

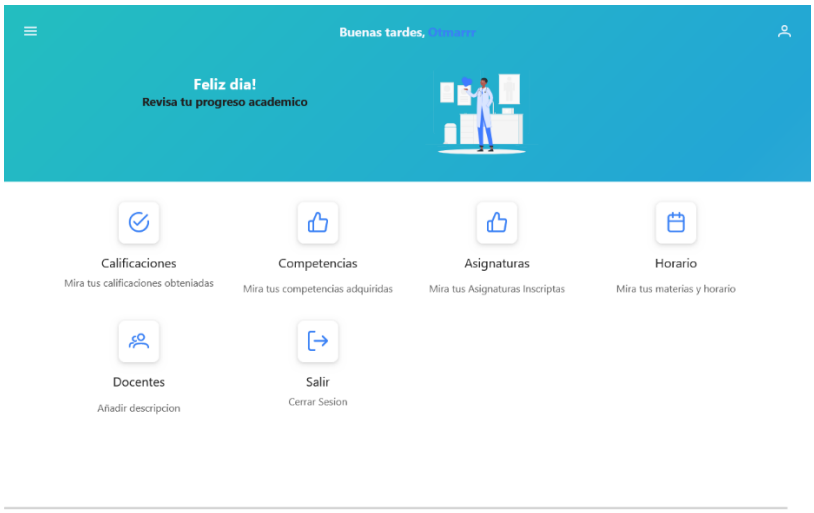
Nombre: Juan Pérez Velázquez

Carnet de Identidad: 12345678



Luego click en ingresar si los datos han sido correctos nos enviar a la página de inicio

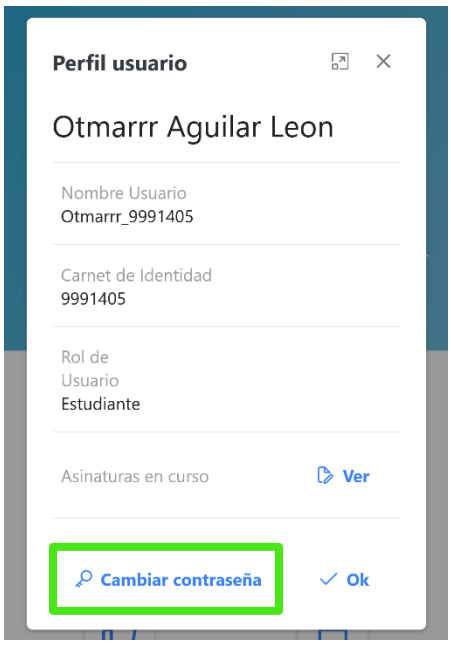




Cambiar Contraseña

Por seguridad se recomienda cambiar su contraseña con los siguientes pasos

1. Ingresar al perfil
2. Boton cambiar Contraseña

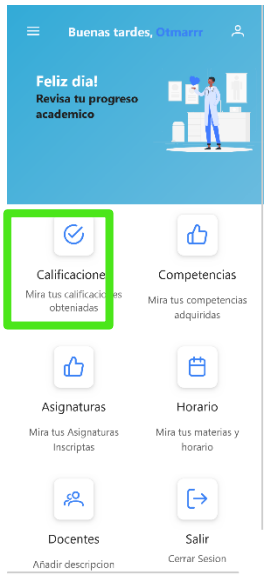


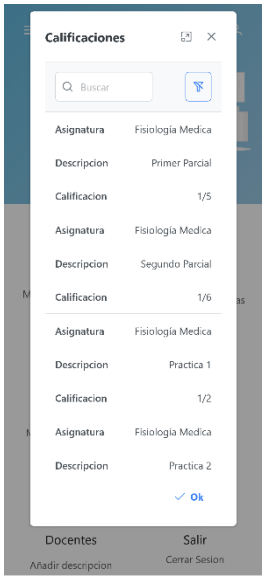
3. Introducir la actual y contraseña y la nueva, pulsa el botón guardar

Vistas del sistema

Calificaciones

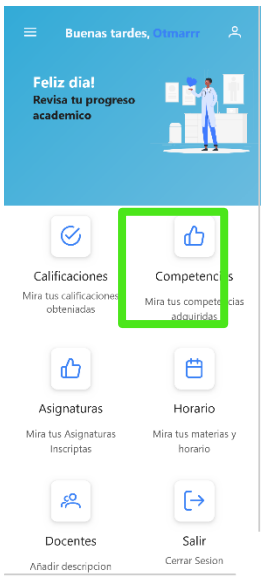
En el módulo de calificaciones se puede ir observando las calificaciones obtenidas de todas las asignaturas en las cuales estén inscritos con los siguientes datos: nombre de la asignatura, descripción de la calificación y la calificación, los datos se pueden filtrar de manera dinámica introduciendo un texto en el cuadro de búsqueda





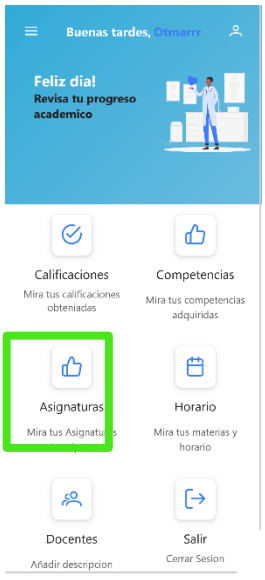
Competencias

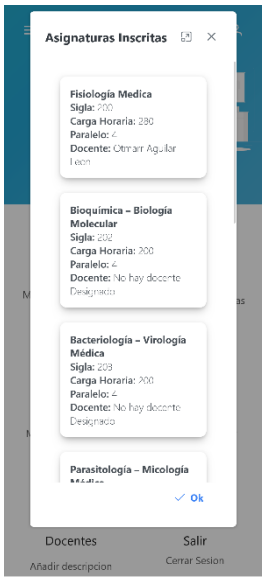
Al pulsar el botón se pudran observar las competencias obtenidas de las diferentes asignaturas con los siguientes datos: nombre de la asignatura, descripción de la competencia y tipo de competencia



Asignaturas

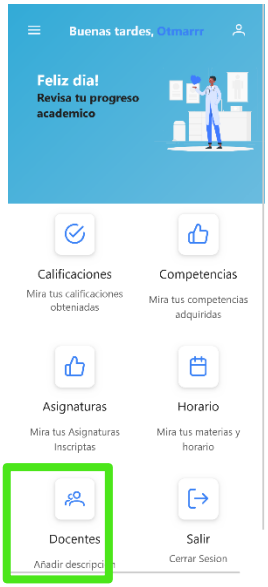
Al pulsar en el botón asignaturas se abrirá una ventana la cual mostrará la información de las asignaturas en las cuales el estudiante está inscrito





Docentes

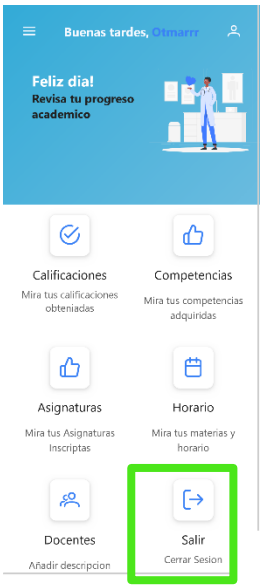
Al pulsar en el botón docentes se abrirá una ventana la cual mostrará la información de los docentes con los cuales esta cursando sus asignaturas, la ventana muestra información de contacto de los docentes como ser: nombre completo, teléfono, correo



Al pulsar la sobre el teléfono se le redireccionara al contacto de WhatsApp correspondiente así mismo con el correo se le redireccionara a Gmail para poder contactar al docente

Salir

Al pulsar sobre el botón Salir se cerrará la sesión y lo redireccionará al login



SISEIN

MANUAL DE USUARIO

ESTUDIANTE

SISTEMA DE CONTROL INDIVIDUAL
ESTUDIANTIL PARA LA VERIFICACIÓN
DE PERFIL PROFESIONAL

Versión 1.0

GUÍA RAPADA DE USO DEL SISTEMA DE CONTROL INDIVIDUAL ESTUDIANTIL PARA LA VERIFICACIÓN DEL PERFIL

¿Qué es el sistema SISEIN?

El sistema SISEIN que implementado en el la carrera de medicina de la universidad publica de el Alto, Nos permite llevar el control del perfil profesional que debe cumplir cada estudiante, verificando e informando la calificaciones y competencias d cada estudiante

Los docentes tienen el control sobre la asignatura la cual desarrollan sus clases ya se creando sus calificaciones como ser practicas y parciales a si mismo pueden adjudicar a los estudiantes las competencias correspondientes a la Asignatura.

Los estudiantes pueden ver sus calificaciones y competencias adquiridas de manera inmediata en el sistema

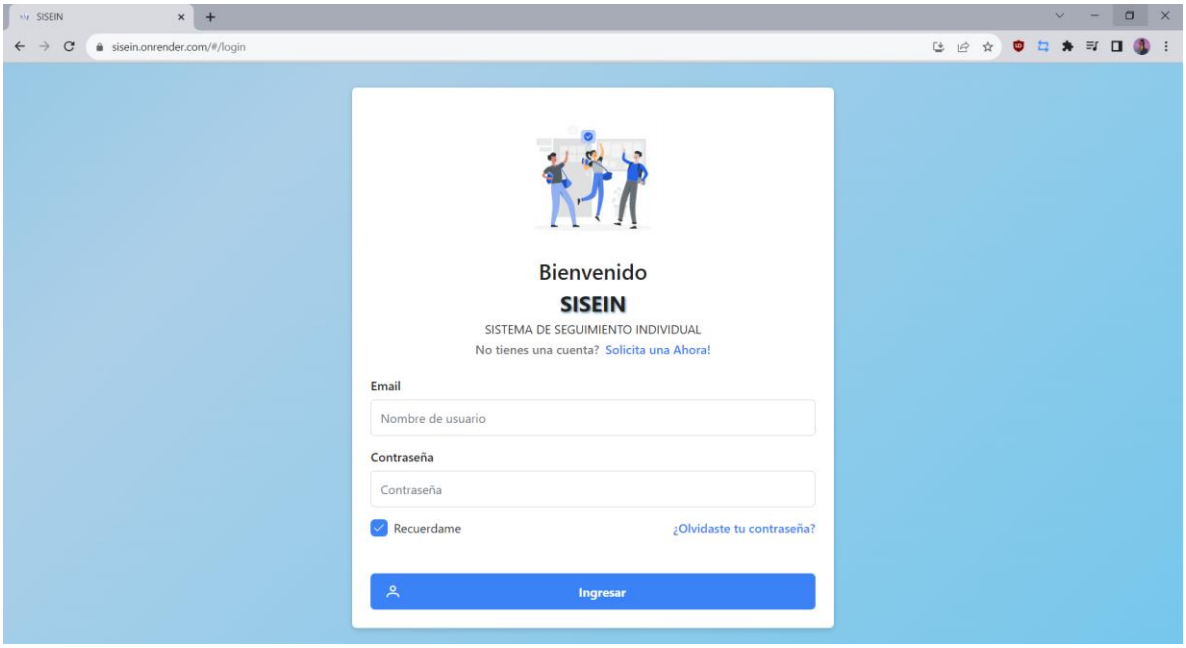
¿Cómo puedo acceder al sistema?

Se puede acceder al sistema desde cualquier navegador de internet (Microsoft Edge, Firefox, Chrome,etc) desde cualquier dispositivo (Computadora, Smartphone ,Tablet) que tenga acceso a internet.

Se puede acceder des la URL:

https://sisein.onrender.com/#/login

Al ingresar al sitio nos redireccionar a la pagina de inicio de sesion



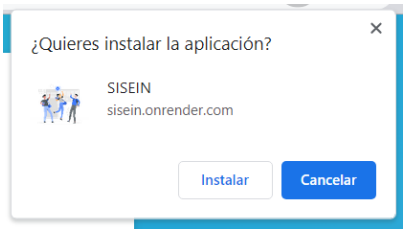
Instalación del sistema (Opcional):

Al momento de ingresar al sistema desde un navegador Chrome nos aparecerá la opción de instalar el sistema en nuestro dispositivo, para ello debemos realizar los siguientes pasos:

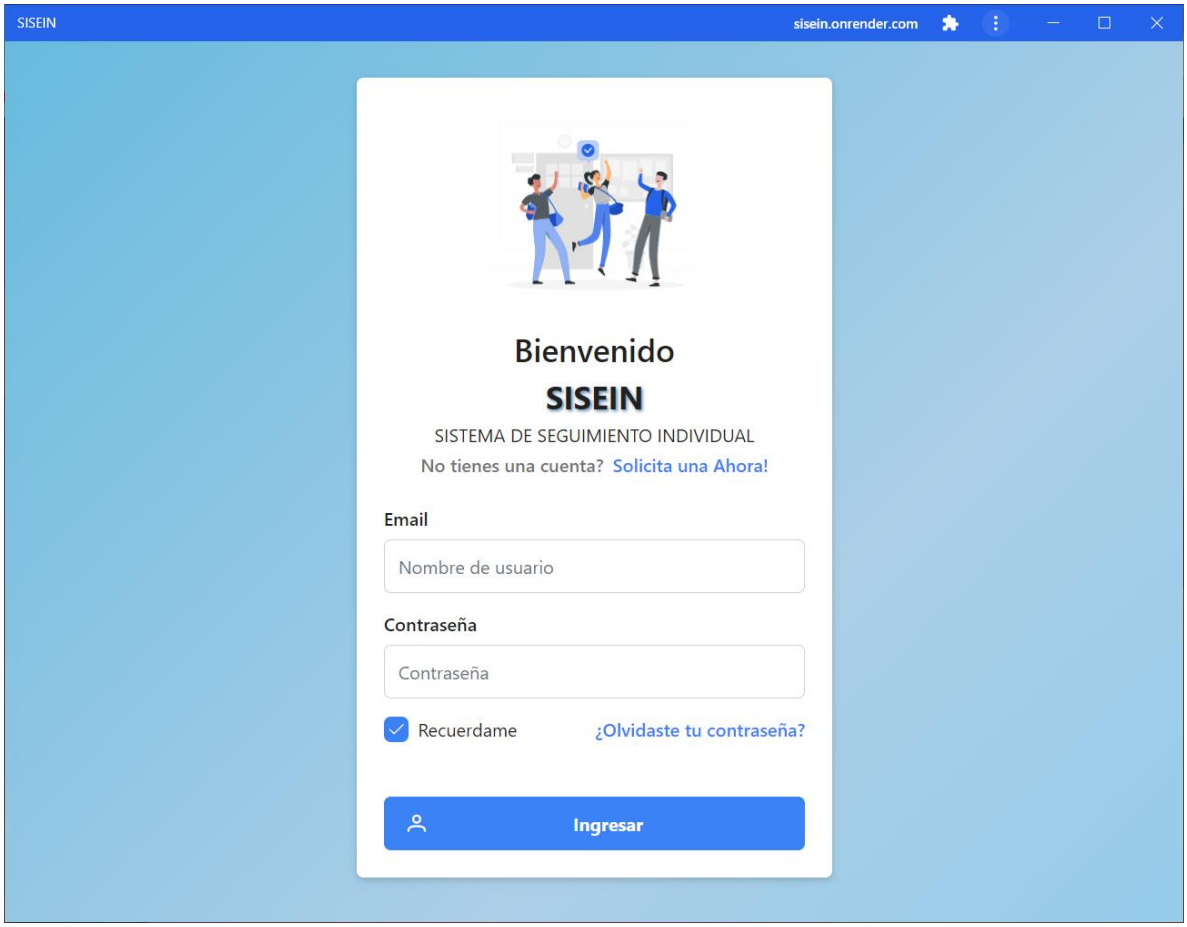
Instalar en Windows

4. Click en botón

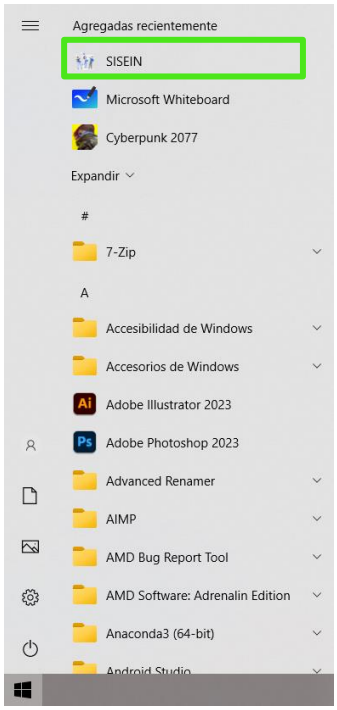
5. Click en Instalar

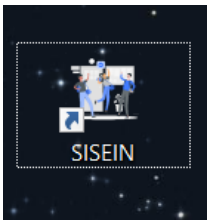


6. Aparece la siguiente ventana



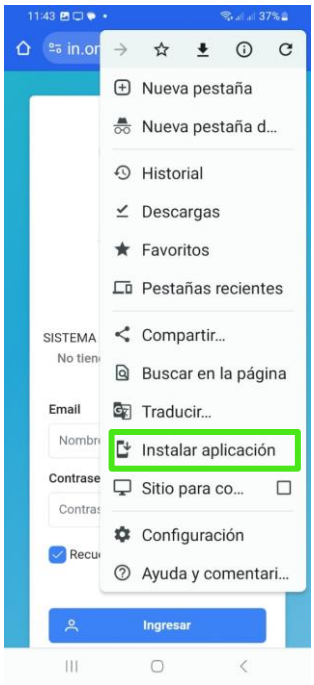
La aplicación se encontrará en tu escritorio y también en tu menú de aplicaciones



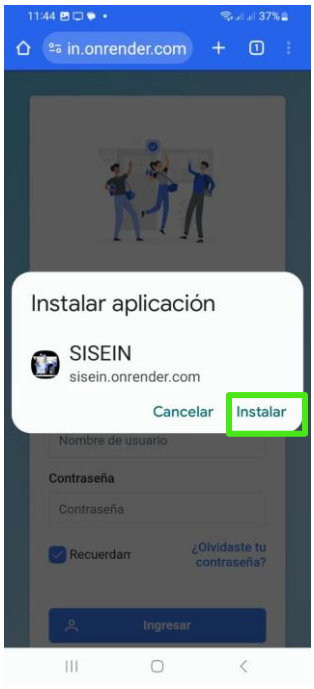


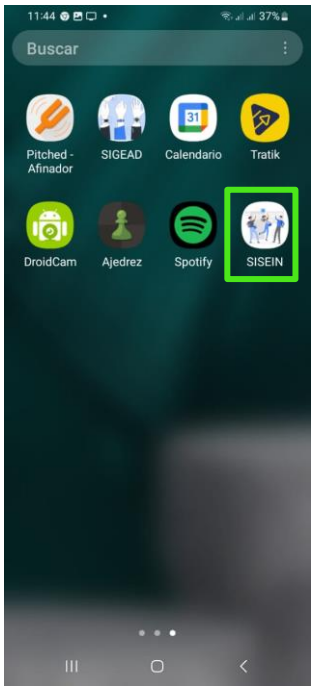
Instalación Android

3. Ingresamos al sistema y hacemos click en el botón mostrar más (tres puntos) del navegados y le damos click en instalar aplicación



4. Click en instalar la esperamos un momento y en el menú de aplicaciones aparece la aplicación instalada





Inicio de sesión

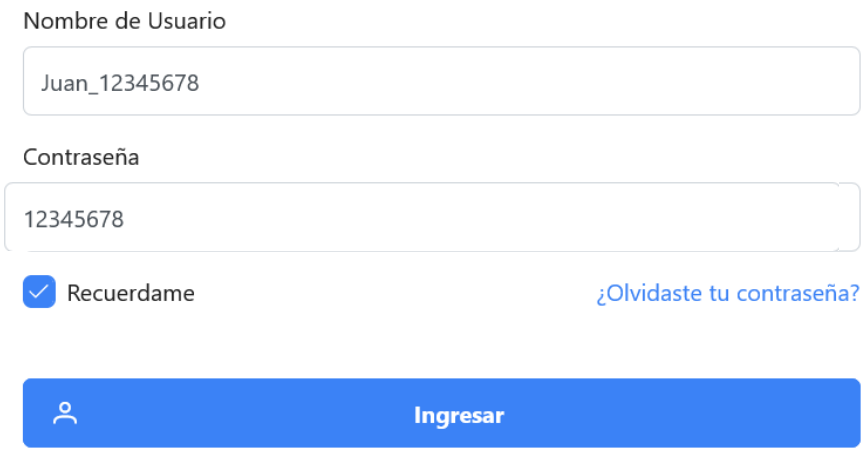
Para iniciar sesión se requiere las credenciales de acceso los cuales son:

Nombre de usuario: PrimerNombre_Carnet de identidad
Contraseña: Carnet de Identidad

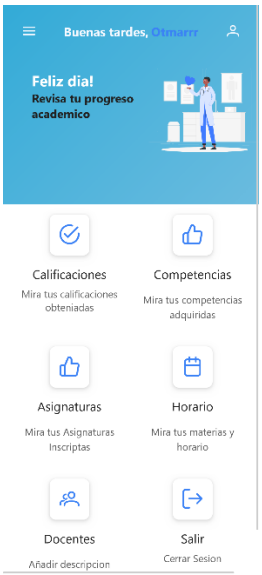
Ejemplo:

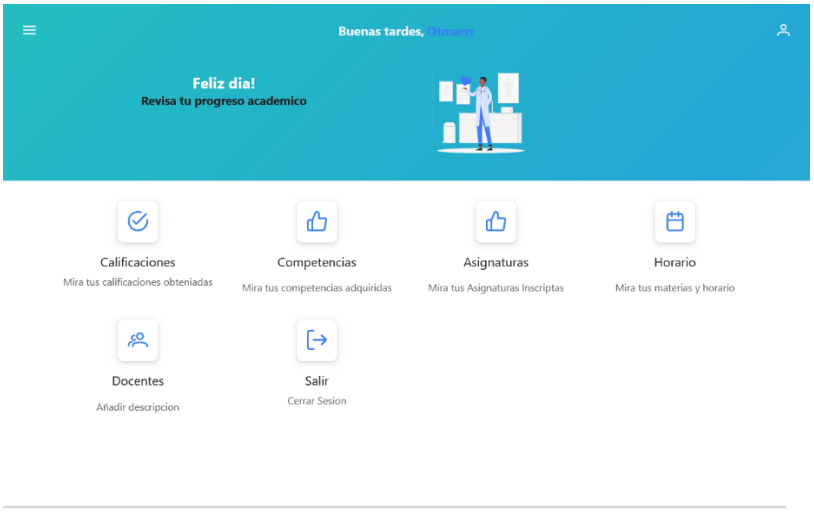
Nombre: Juan Pérez Velázquez

Carnet de Identidad: 12345678



Luego click en ingresar si los datos han sido correctos nos enviar a la pagina de inicio

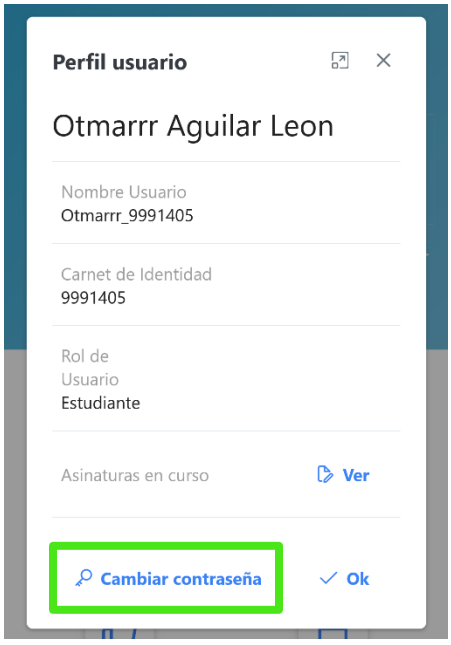




Cambiar Contraseña

Por seguridad se recomienda cambiar su contraseña con los siguientes pasos

4. Ingresar al perfil
5. Boton cambiar Contraseña

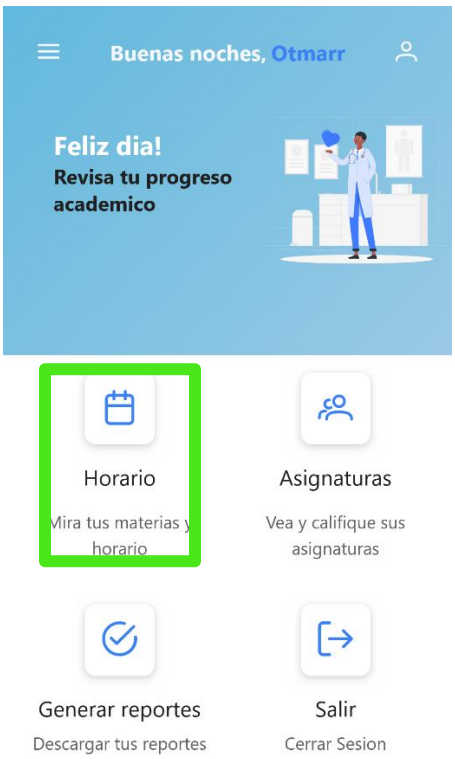


6. Introducir la actual y contraseña y la nueva, pulsa el botón guardar

Vistas del sistema

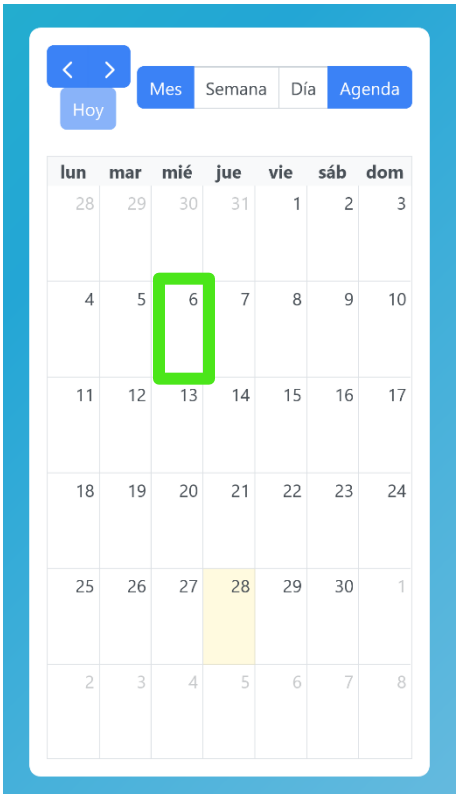
Horario

El módulo de horario se puede ver y agendar actividades en un calendario interactivo

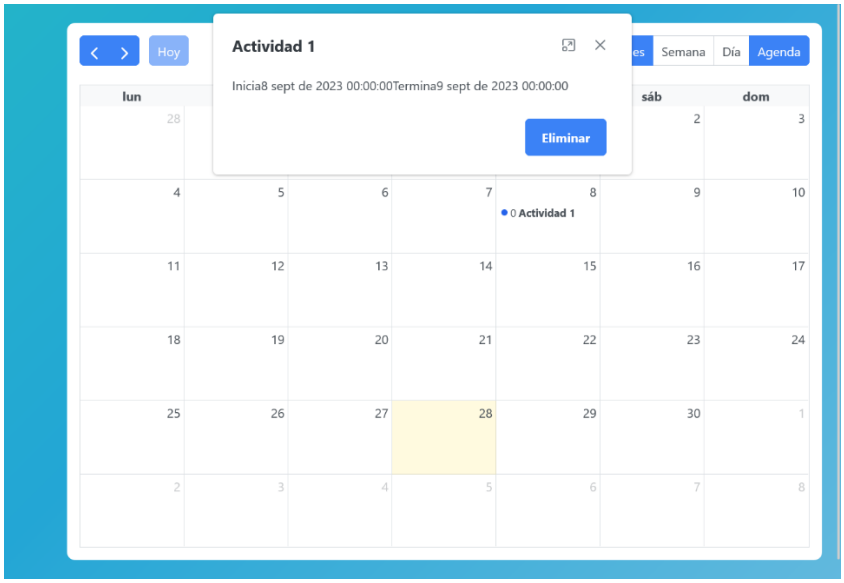


Para crea un nuevo evento o actividad en el calendario seguir los siguientes pasos

1. Pulsar sobre la fecha en la que quiere agendar, si la actividad requiere varios días arrastre el mouse o dedo sobre los días que requiere
2. Se abrirá una ventana en la cual debe insertar una descripción de la actividad también puede modificar la fecha
3. Presionar en botón enviar

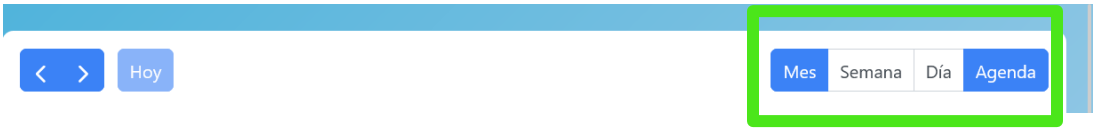


Para ver el evento presionar sobre el evento creado



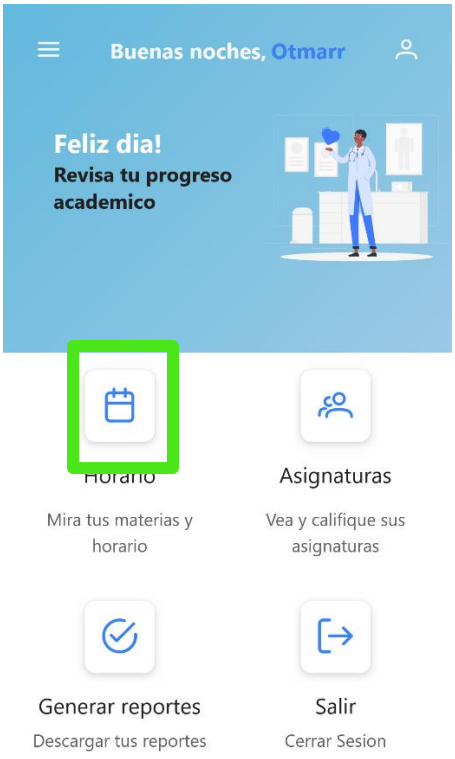
Puede eliminar el evento presionando el botón “Eliminar”

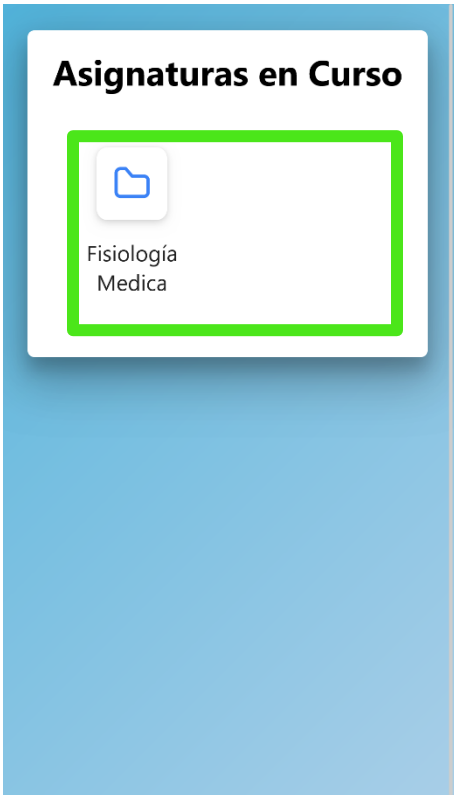
El calendario se puede visualizar de diferentes maneras por Mes, semana, día y agenda



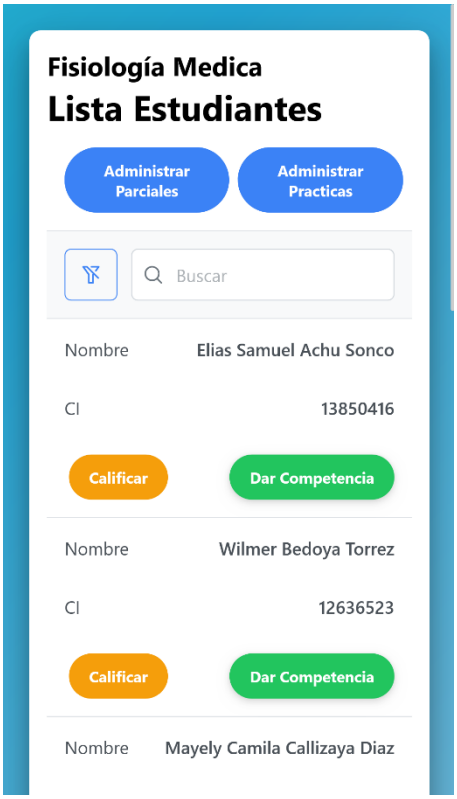
Módulo de asignaturas

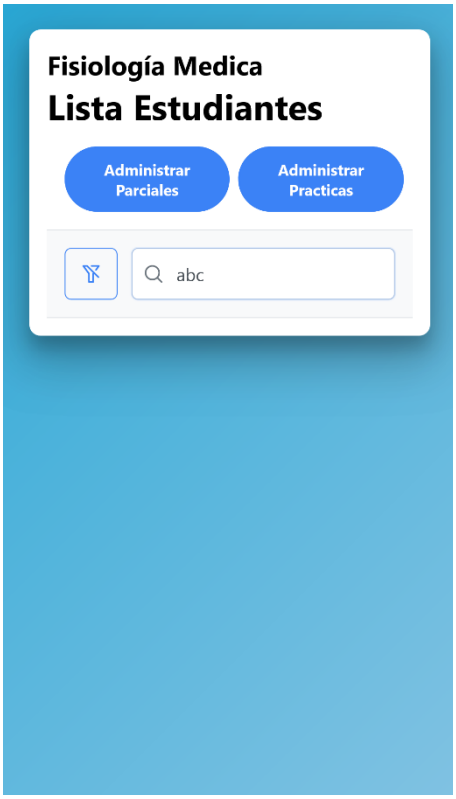
En el siguiente modulo se puede administrar las asignaturas las cuales se están designadas al docente, en las asignaturas en curso se van listando las asignaturas en las cuales están designadas al docente





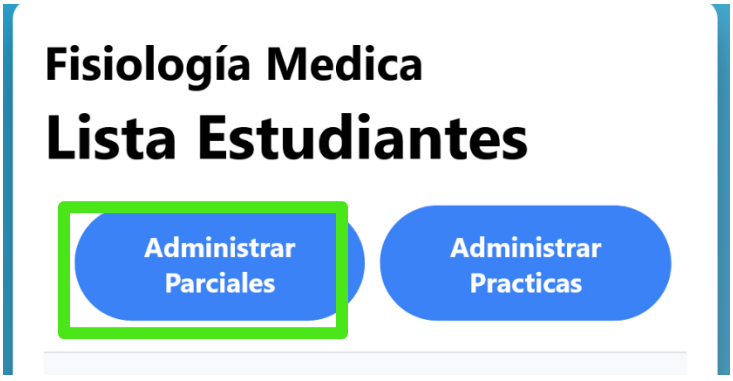
Al ingresar a una de las asignaturas listadas se desplegará la lista de los estudiantes inscritos en la dicha asignatura, ingresando datos en el cuadro de búsqueda se pueden filtrar los estudiante

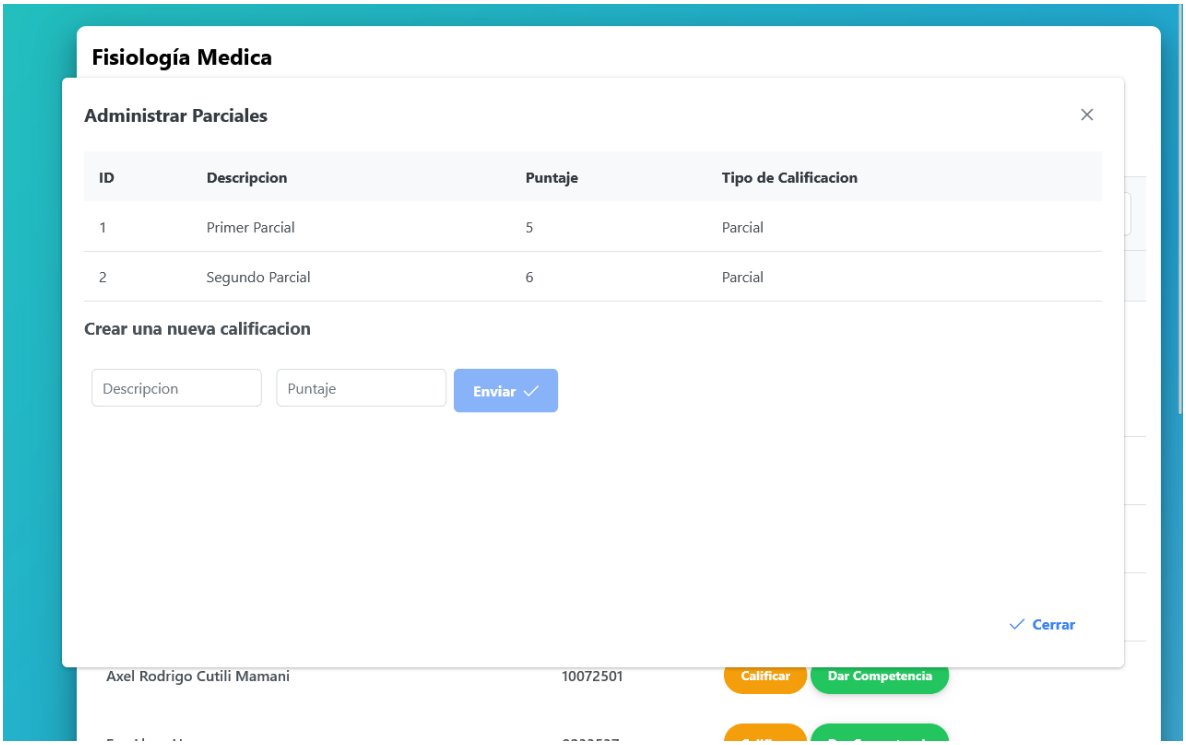




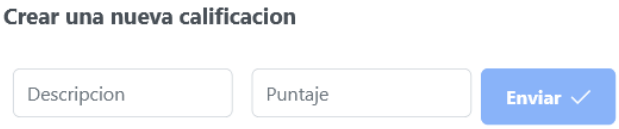
Administrar Parciales

Al presionar administrar parciales se desplegará una ventana





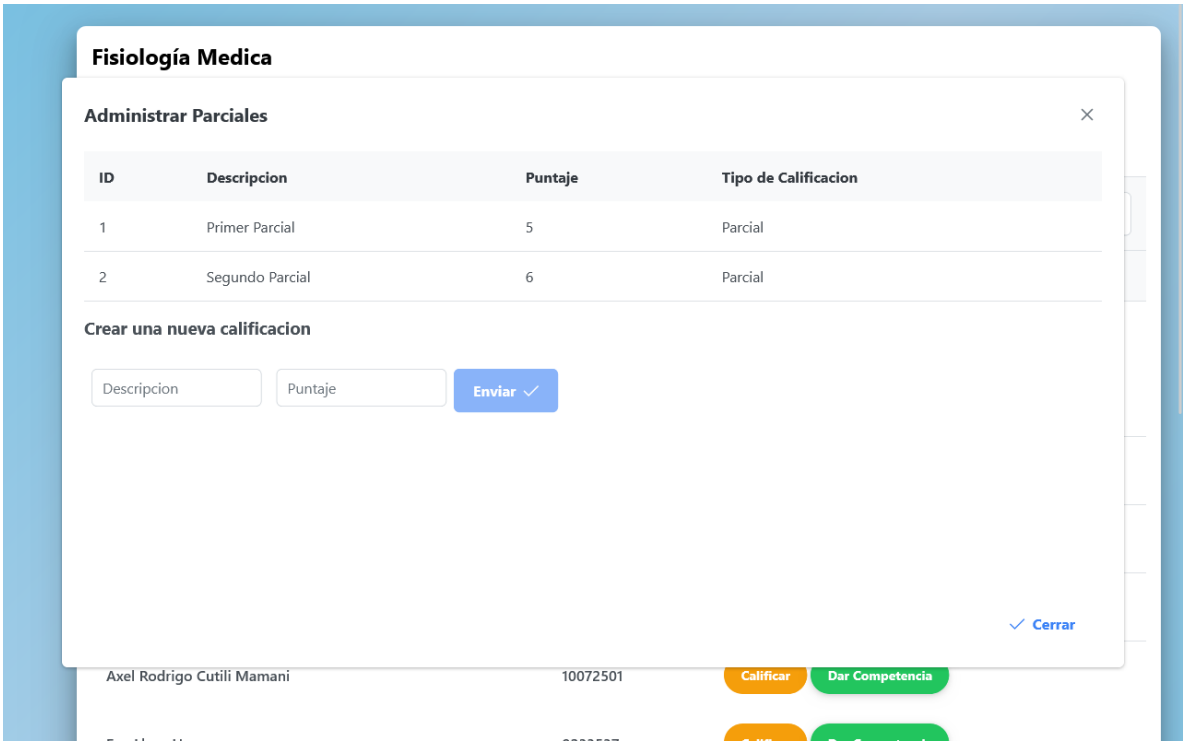
En la anterior pantalla podemos ver la lista de las calificaciones creadas y también podemos crear una nueva



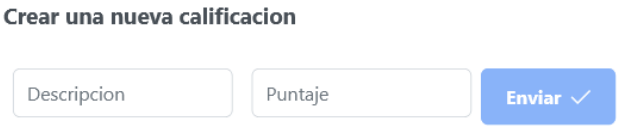
Para crear una calificación debe rellenar una descripción de la calificación y el puntaje sobre cuál será calificada el parcial

Administrar Parciales

Al presionar administrar practicas se desplegará una ventana



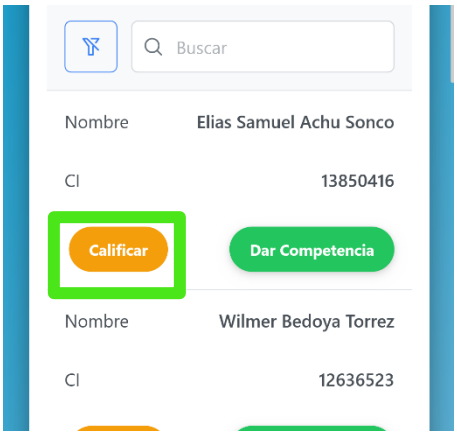
Para crear una calificación debe rellenar una descripción de la práctica y el puntaje sobre cuál será calificada la practica



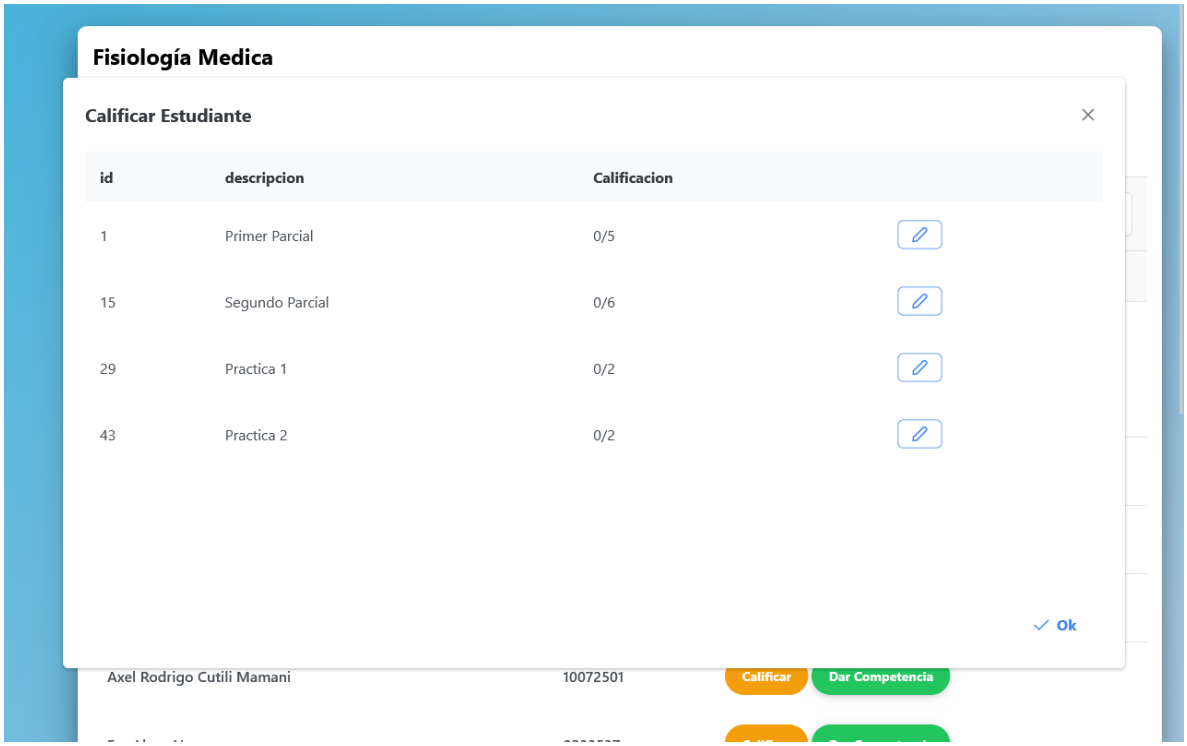
Calificar

Una vez creada las calificaciones o las practicas para poder calificar al estudiante se deben seguir los siguientes pasos:

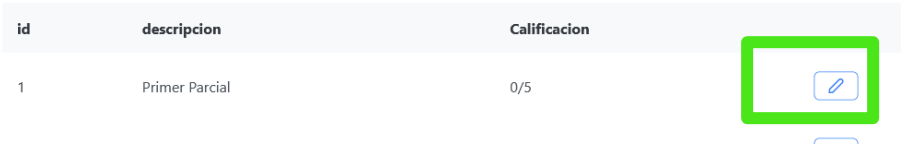
1. Presionar sobre Calificar



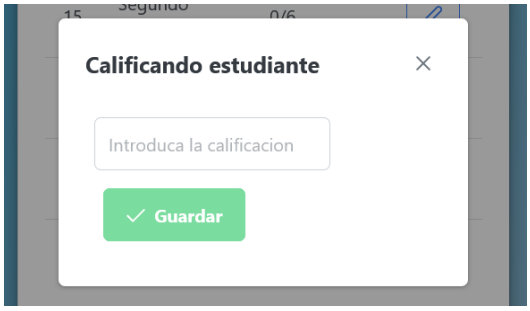
2. Se abrirá la ventana donde aparecerán las calificaciones creadas

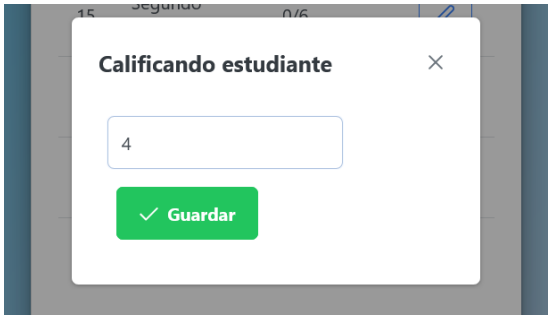


3. Presionar el botón calificar



4. Introducir la calificación



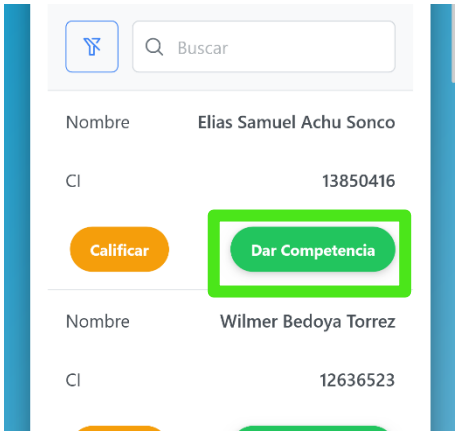


La calificación no puede superar la nota definida a la hora de crear la calificación o no dejara ingresar la calificación

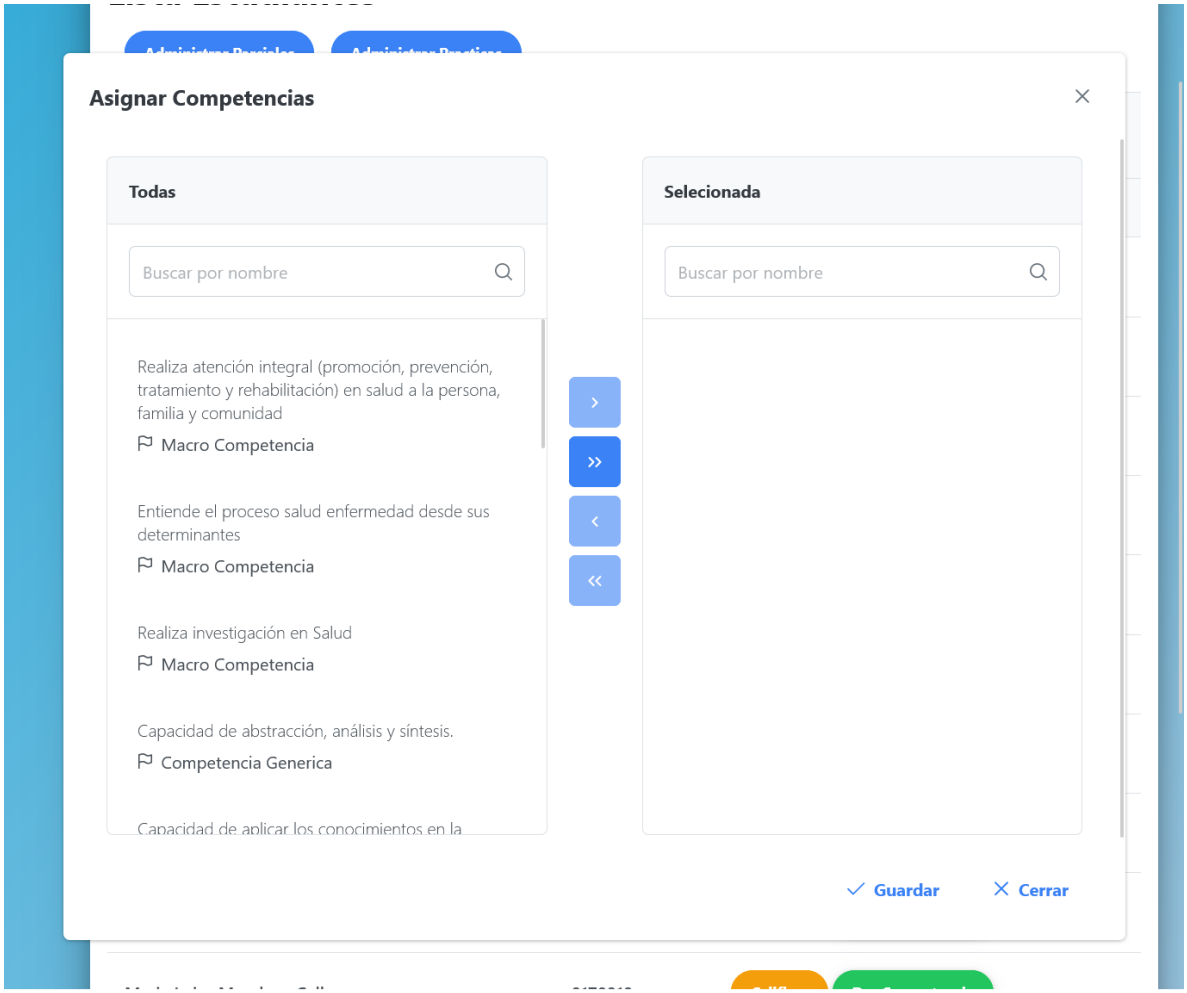
Adjudicar Competencia

Para adjudicar la competencia se deben seguir los siguientes pasos:

1. Presionar sobre el botón dar competencia

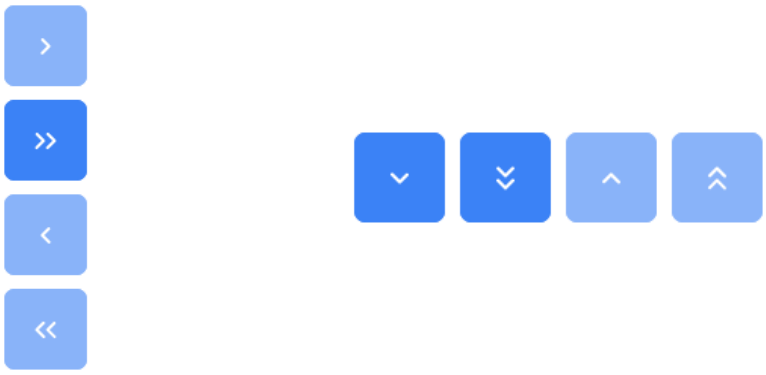


2. En la siguiente venta puede arrastrar o usar las flechas para asignar las competencias de la asignatura al estudiante.
   La competencia de la izquierda son la competencia de la asignatura, las competencias de la de derecha son la competencia que tiene el estudiante en la asignatura en la cual se encuentra

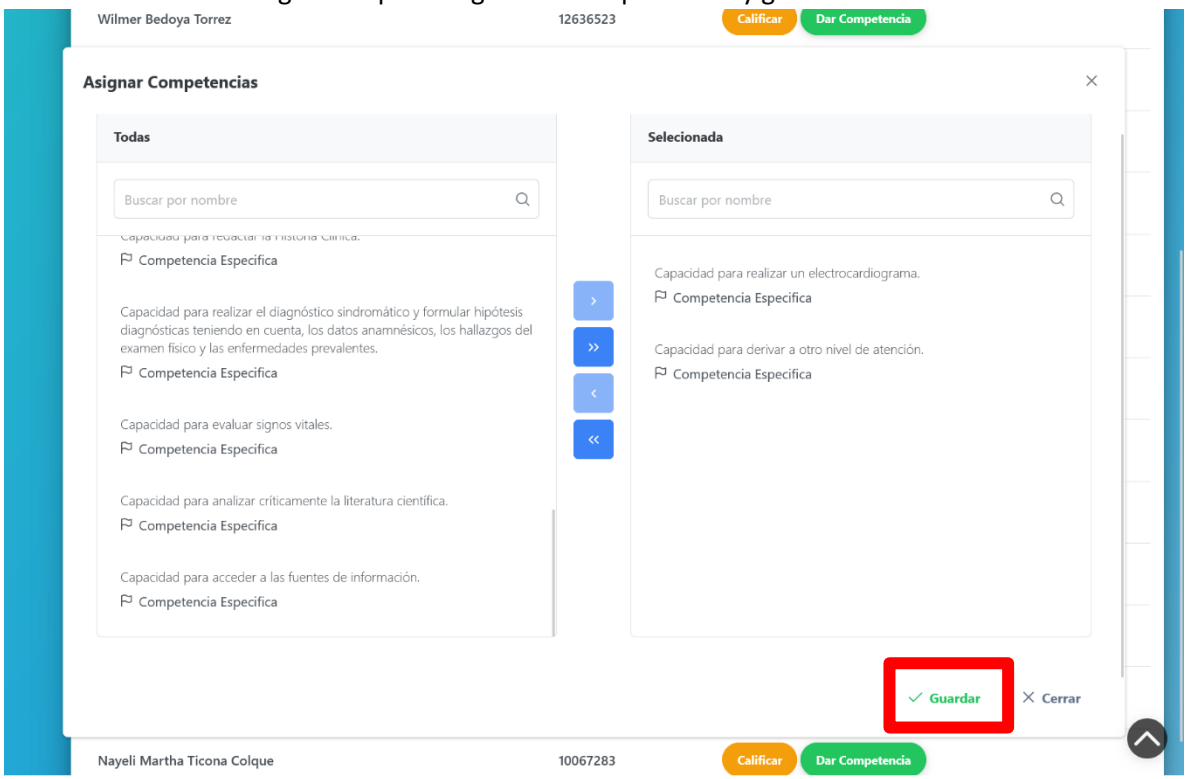


Las flechas de en medio pueden llevar las competencias de derecha a izquierda y viceversa

Al pulsar sobre una o varias competencias se habilita la flecha simple la cual llevara la competencia a la izquierda, las flechas dobles sirven para llevar todas las competencias.

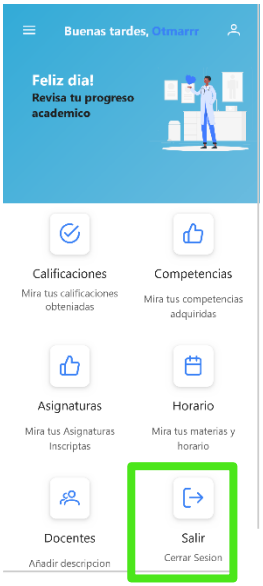


3. Presionar el botón guardar para asignar las competencias y guardarlas definitivamente



Salir

Al pulsar sobre el botón Salir se cerrará la sesión y lo redireccionará al login


Buenas tardes, Otmarrr
Feliz dia!
Revisa tu progreso academico
Calificaciones
Mira tus calificaciones obteniadas
Competencias
Mira tus competencias adquiridas
Asignaturas
Mira tus Asignaturas Inscriptas
Horario
Mira tus materias y horario
Docentes
Añadir descripcion
Salir
Cerrar Sesion

SISEIN

MANUAL DE TÉCNICO

SISTEMA DE CONTROL INDIVIDUAL
ESTUDIANTIL PARA LA VERIFICACIÓN
DE PERFIL PROFESIONAL

Versión 1.0

1. Introducción

En este manual se describa los pasos necesarios para poner en funcionamiento el aplicativo, se requiere que la persona encargada de la instalación, tenga conocimientos básicos de sistemas.

2. Objetivo General

Detallar la información necesaria para realizar la instalación, configuración y puesta en funcionamiento del aplicativo.

2.1. Objetivos Específicos

- Definir claramente el proceso de instalación del aplicativo.
- Detallar los requerimientos mínimos de hardware y software para el funcionamiento del aplicativo.
- Describir las herramientas utilizadas en el desarrollo del aplicativo.

3. Requerimientos Técnicos

Una de las herramientas necesarias al momento de realizar desarrollo web es una computadora la cual debería cumplir con las siguientes características:

- Procesador: Intel i5 o AMD Ryzen 5 en adelante.
- Memoria RAM: 16 GB.
- Almacenamiento: disco SSD (estado sólido).
- Conexión a internet.

Sin embargo, estos requisitos son flexibles.

Para la implementación del proyecto deberá contar con el hardware recomendable de:

- 2 Cores 3.2 GHz. o superior
- 2 Gb de RAM.
- 16 Gb de almacenamiento.

Para el uso del software desarrollado los equipos o dispositivos deberán cumplir con las siguientes características:

- Procesador: Dual Core o superior
- Memoria RAM: 2 Gb o superior
- Almacenamiento: 120 GB o mas
- Sistema Operativo: Windows, Linux, Mac
- Otros: Conexión a internet

4. Herramientas utilizadas para el desarrollo

4.1. Nestjs

Nest.js es un framework de desarrollo web basado en Node.js que utiliza TypeScript para proporcionar una estructura de programación sólida y altamente escalable.

Se inspira en otros frameworks de desarrollo populares, como Angular y Spring. Y se centra en la creación aplicaciones de Backend altamente escalables y modulares, aunque también puedes desarrollar aplicaciones web.

Es compatible e incluye integraciones por defecto con una amplia variedad de bases de datos y sistemas de autenticación. Además, se puede utilizar con una amplia variedad de módulos de terceros para añadir funcionalidades adicionales. También se integra con herramientas de testing y depuración para ayudar a probar y depurar el código de la aplicación.

4.2. Angular

Angular es un framework para aplicaciones web desarrollado en TypeScript, de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página. Su objetivo es aumentar las aplicaciones basadas en navegador con capacidad de Modelo Vista Controlador (MVC), en un esfuerzo para hacer que el desarrollo y las pruebas sean más fáciles.

La biblioteca lee el HTML que contiene atributos de las etiquetas personalizadas adicionales, entonces obedece a las directivas de los atributos personalizados, y une las piezas de entrada o salida de la página a un modelo representado por las variables estándar de JavaScript.

Angular se basa en clases tipo "Componentes", cuyas propiedades son las usadas para hacer el binding de los datos. En dichas clases tenemos propiedades (variables) y métodos (funciones a llamar).

Angular es la evolución de AngularJS aunque incompatible con su predecesor.

4.3. PrimeNG

PrimeNG es una biblioteca de componentes de interfaz de usuario de código abierto para Angular. Ofrece una amplia variedad de componentes como tablas, menús, botones, calendarios, entre otros, que ayudan a los desarrolladores a crear aplicaciones web atractivas y fáciles de usar.

4.4. MariaDB

MariaDB es un sistema de gestión de bases de datos relacionales (RDBMS) gratuito y de código abierto. Fue creado por los desarrolladores originales de MySQL por la preocupación de que MySQL pasara a ser comercializado después de que Oracle lo adquiriera en 2009.

MariaDB está escrito en C y C++ y es compatible con varios lenguajes de programación, incluidos C, C#, Java, Python, PHP y Perl. MariaDB también es compatible con todos los principales sistemas operativos, incluidos Windows, Linux y macOS.

4.5. Visual studio Code

Visual Studio Code (VS Code) es un editor de código fuente desarrollado por Microsoft. Es software libre y multiplataforma, está disponible para Windows, GNU/Linux y macOS. VS Code tiene una buena integración con Git, cuenta con soporte para depuración de código, y dispone de un sinnúmero de extensiones, que básicamente te da la posibilidad de escribir y ejecutar código en cualquier lenguaje de programación.

4.6. NodeJs

Node.js®, Node.js, es un entorno en tiempo de ejecución multiplataforma para la capa del servidor (en el lado del servidor) basado en JavaScript.

Node.js es un entorno controlado por eventos diseñado para crear aplicaciones escalables, permitiéndote establecer y gestionar múltiples conexiones al mismo tiempo. Gracias a esta característica, no tienes que preocuparte con el bloqueo de procesos, pues no hay bloqueos.

4.6.1. NPM

npm es parte esencial de Node.js, el entorno de ejecución de javaScript en el lado del servidor basado en el motor V8 de Google. Es muy seguramente la principal razón del gran éxito de Node permitiendo que cientos de desarrolladores puedan compartir paquetes de software entre distintos proyectos.

En este momento, se han publicado 1,493,231 paquetes por medio de su repositorio con más de 27 mil millones de descargas.

npm responde a las siglas de Node Package Manager o manejador de paquetes de node, es la herramienta por defecto de JavaScript para la tarea de compartir e instalar paquetes.

5. Instalación del backend

Para realizar la instalación se debe copiar el código fuente el cual se encuentra en cd adjunto a este manual, una vez copiado la carpeta contenida en la carpeta backend procedemos a introducir el siguiente comando en una consola

- npm install

Para el desarrollo

- npm run start:dev

Para producción

- npm run start:prod

5.1. Configuración de la Base de datos

Para conectar la base de datos MariaDB o MySql al Servidor Backend se debe contar con las credenciales de acceso a la base de datos, con esos datos se procede a crear el archivo .env dentro de la carpeta del proyecto con la siguiente estructura

- DATABASE_HOST
- DATABASE_PORT
- DATABASE_USER
- DATABASE_PASSWORD
- DATABASE_NAME
- DATABASE_SSL

6. Instalación Frontend

Para realizar la instalación se debe copiar el código fuente el cual se encuentra en cd adjunto a este manual, una vez copiado la carpeta contenida en la carpeta frontend procedemos a introducir el siguiente comando en una consola

- npm install

Para el desarrollo

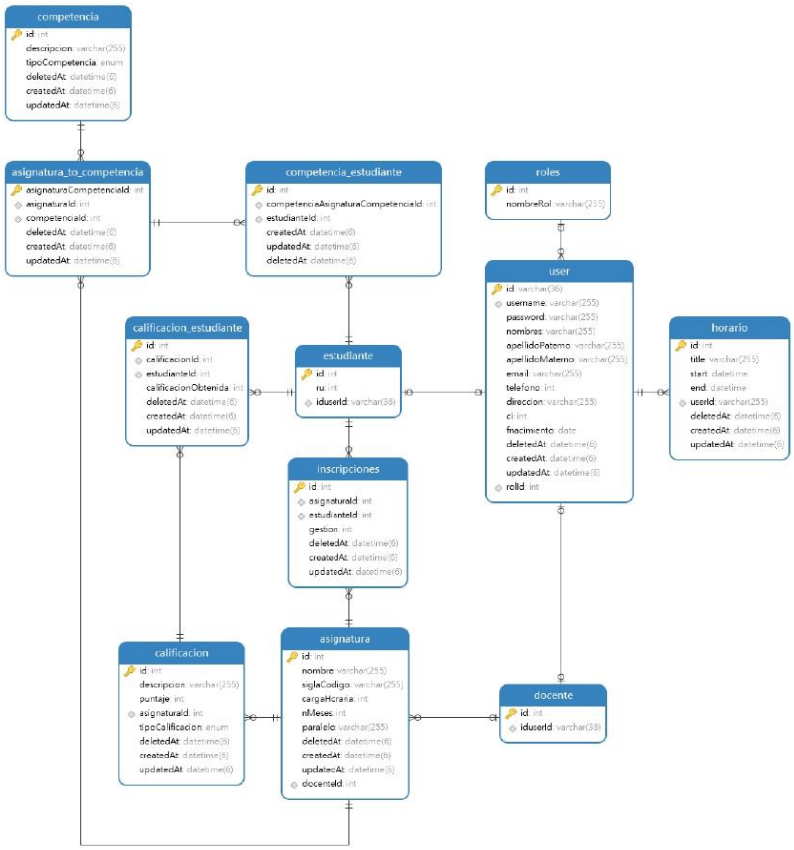
- ng serve

Para producción

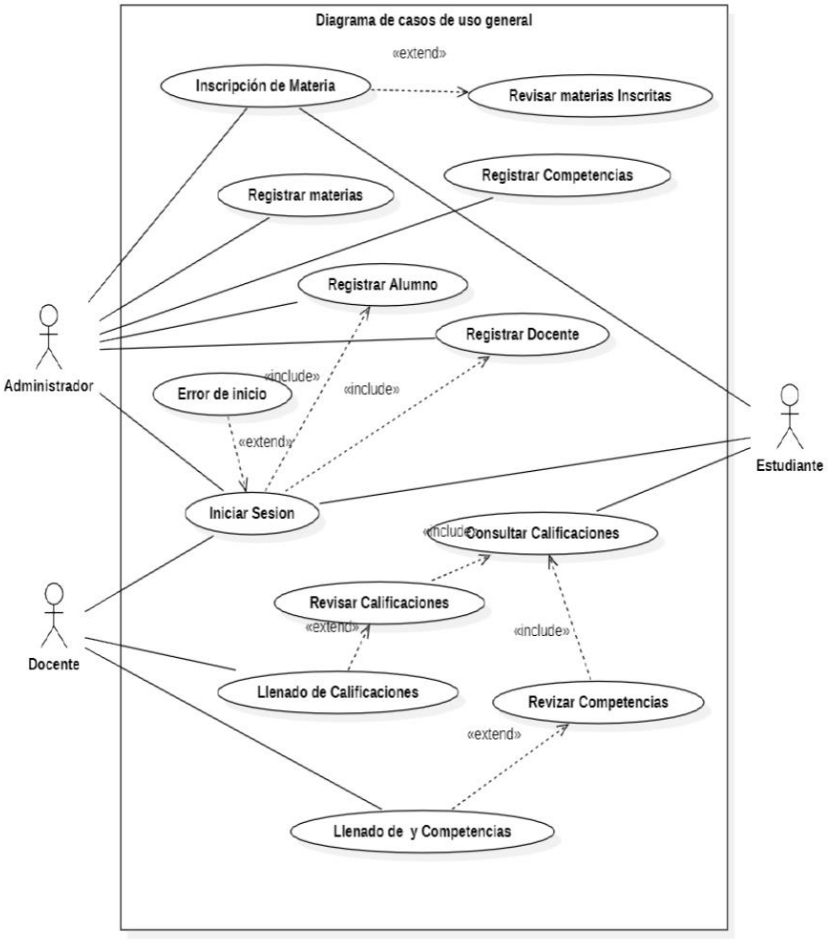
- ng build

una vez ejecutado el código para producción dentro la carpeta del proyecto se crea la carpeta “dist” la cual es el condigo compilado para subirlo a un servidor web.

7. Modelo Físico de la Base de Datos



8. Diagrama general de casos de uso



SISEIN

MANUAL DE ADMINISTRACIÓN

SISTEMA DE CONTROL INDIVIDUAL ESTUDIANTIL PARA LA VERIFICACIÓN DE PERFIL PROFESIONAL

Versión 1.0

GUÍA RAPADA DE USO DEL SISTEMA DE CONTROL INDIVIDUAL ESTUDIANTIL PARA LA VERIFICACIÓN DEL PERFIL

¿Qué es el sistema SISEIN?

El sistema SISEIN que implementado en el la carrera de medicina de la universidad publica de el Alto, Nos permite llevar el control del perfil profesional que debe cumplir cada estudiante, verificando e informando la calificaciones y competencias d cada estudiante

Los docentes tienen el control sobre la asignatura la cual desarrollan sus clases ya se creando sus calificaciones como ser practicas y parciales a si mismo pueden adjudicar a los estudiantes las competencias correspondientes a la Asignatura.

Los estudiantes pueden ver sus calificaciones y competencias adquiridas de manera inmediata en el sistema

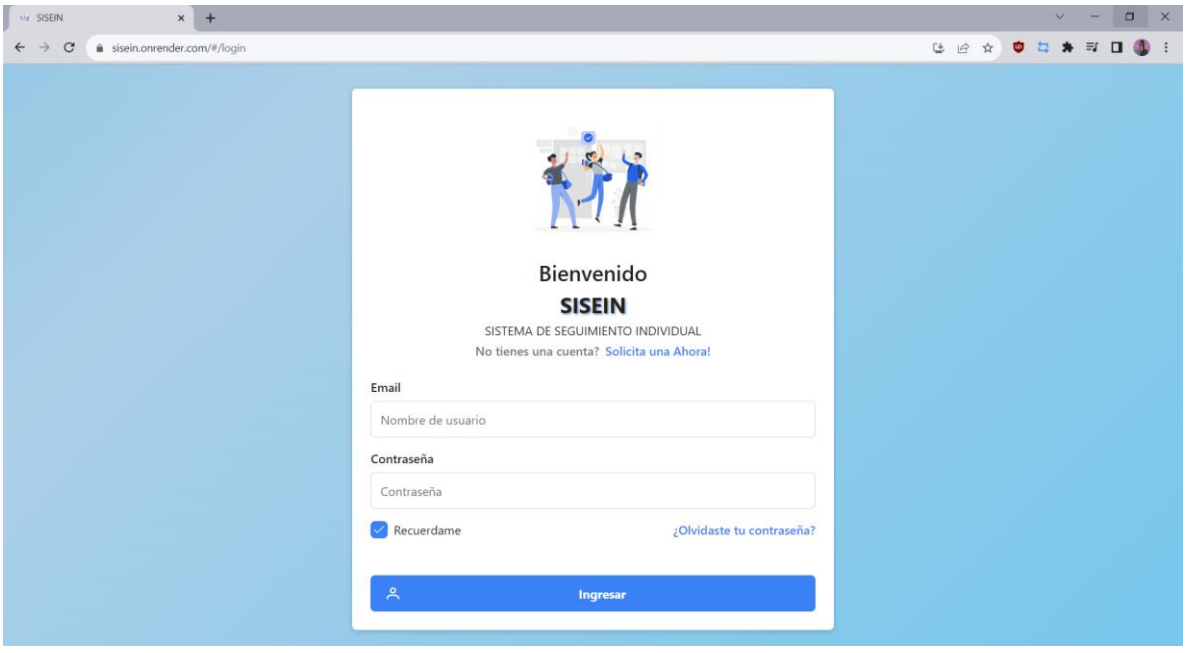
¿Cómo puedo acceder al sistema?

Se puede acceder al sistema desde cualquier navegador de internet (Microsoft Edge, Firefox, Chrome,etc) desde cualquier dispositivo (Computadora, Smartphone ,Tablet) que tenga acceso a internet.

Se puede acceder des la URL:

https://sisein.onrender.com/#/login

Al ingresar al sitio nos redireccionar a la pagina de inicio de sesion



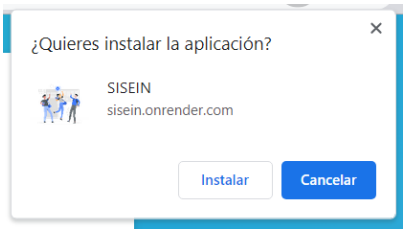
Instalación del sistema (Opcional):

Al momento de ingresar al sistema desde un navegador Chrome nos aparecerá la opción de instalar el sistema en nuestro dispositivo, para ello debemos realizar los siguientes pasos:

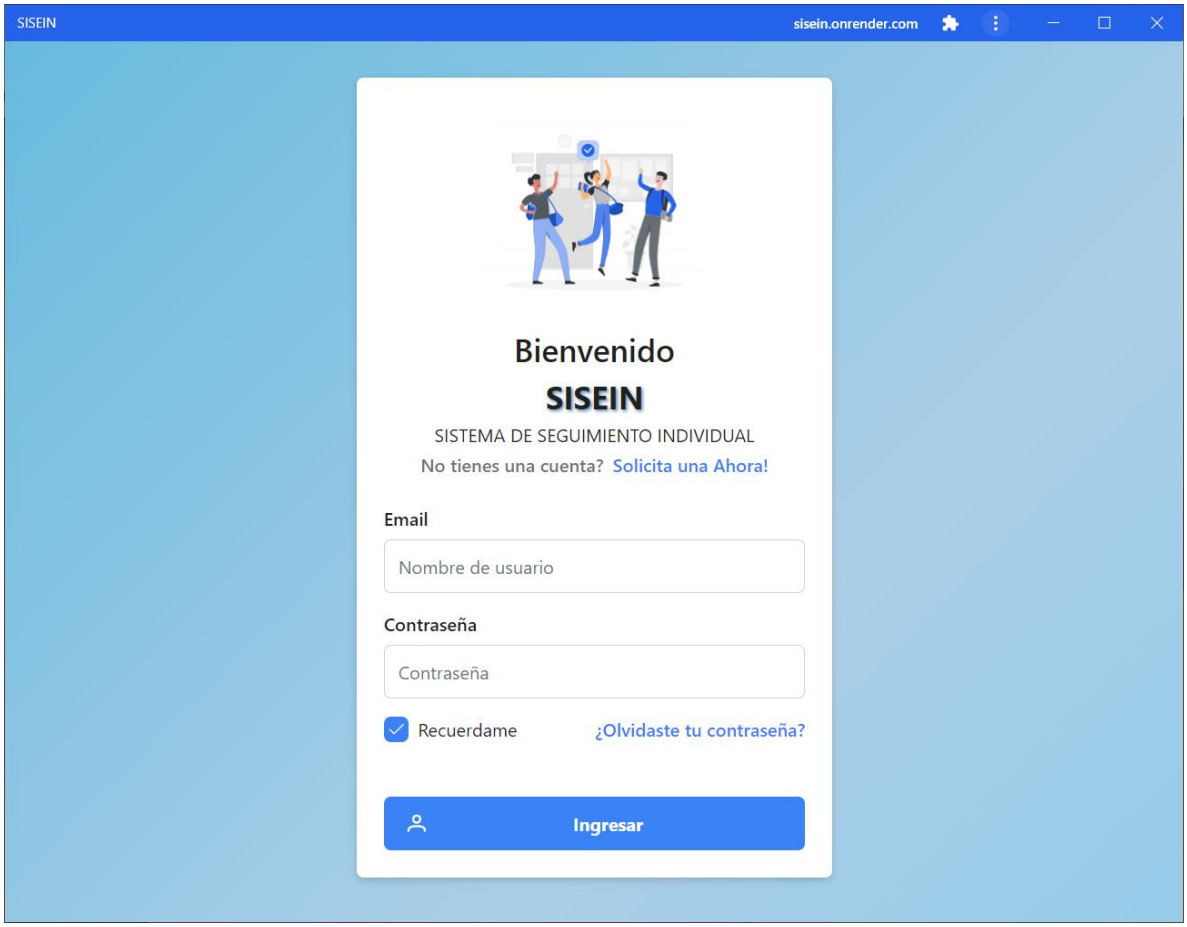
Instalar en Windows

7. Click en botón

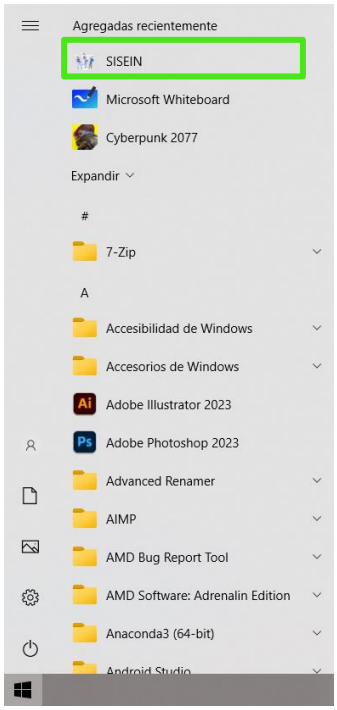
8. Click en Instalar

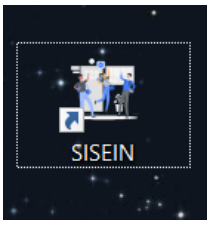


9. Aparece la siguiente ventana



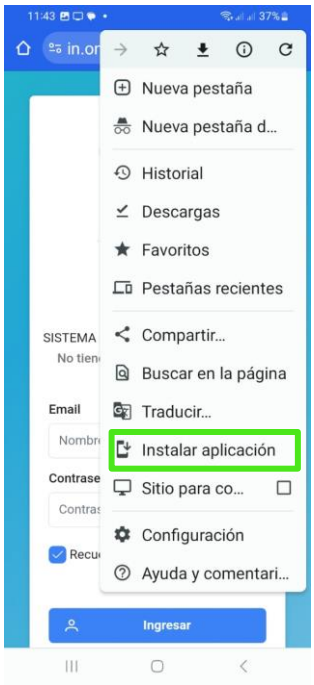
La aplicación se encontrará en tu escritorio y también en tu menú de aplicaciones



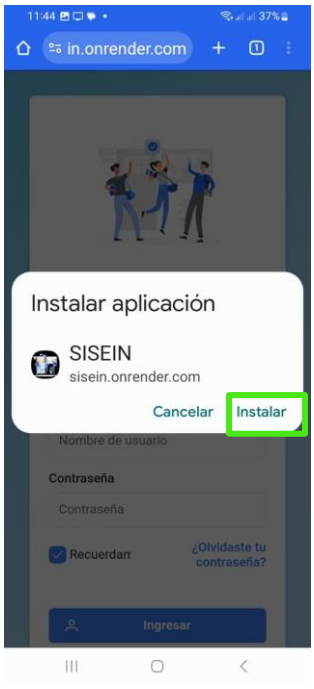


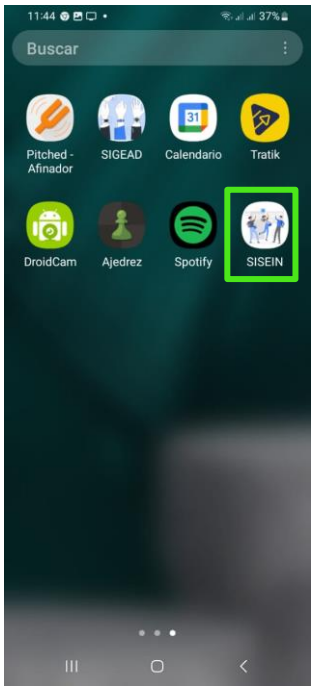
Instalación Android

5. Ingresamos al sistema y hacemos click en el botón mostrar más (tres puntos) del navegados y le damos click en instalar aplicación



6. Click en instalar la esperamos un momento y en el menú de aplicaciones aparece la aplicación instalada





Inicio de sesión

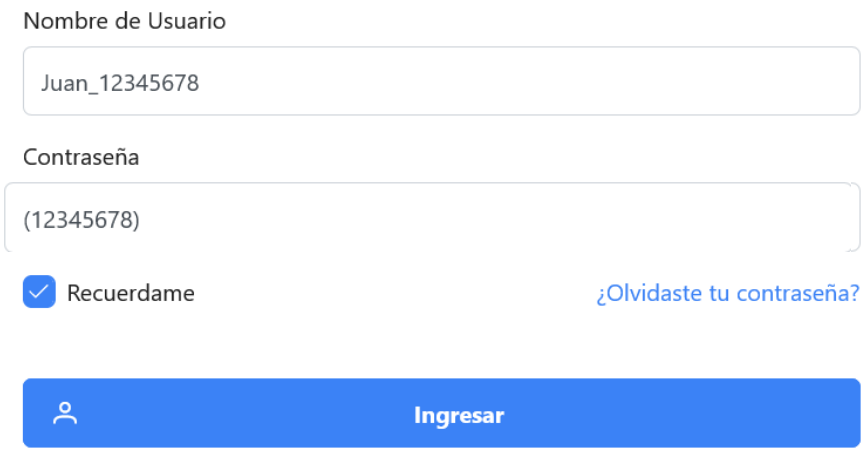
Para iniciar sesión se requiere los credenciales de acceso los cuales son :

Nombre de usuario: PrimerNombre_Carnet de identidad
Contraseña: Carnet de Identidad

Ejemplo:

Nombre: Juan Pérez Velázquez

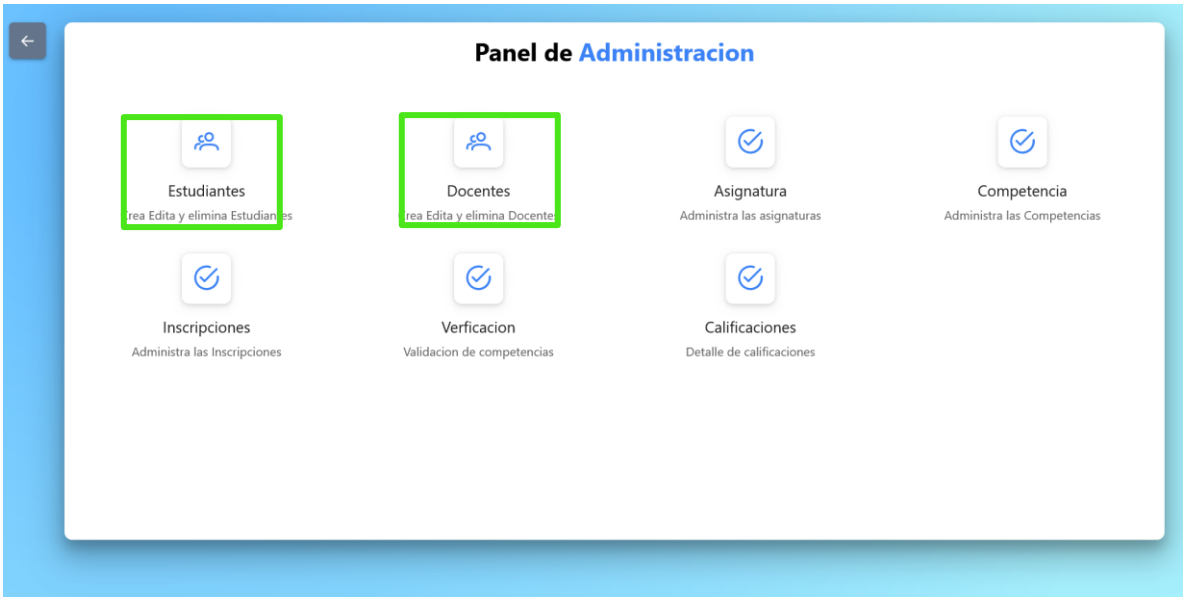
Carnet de Identidad: 12345678



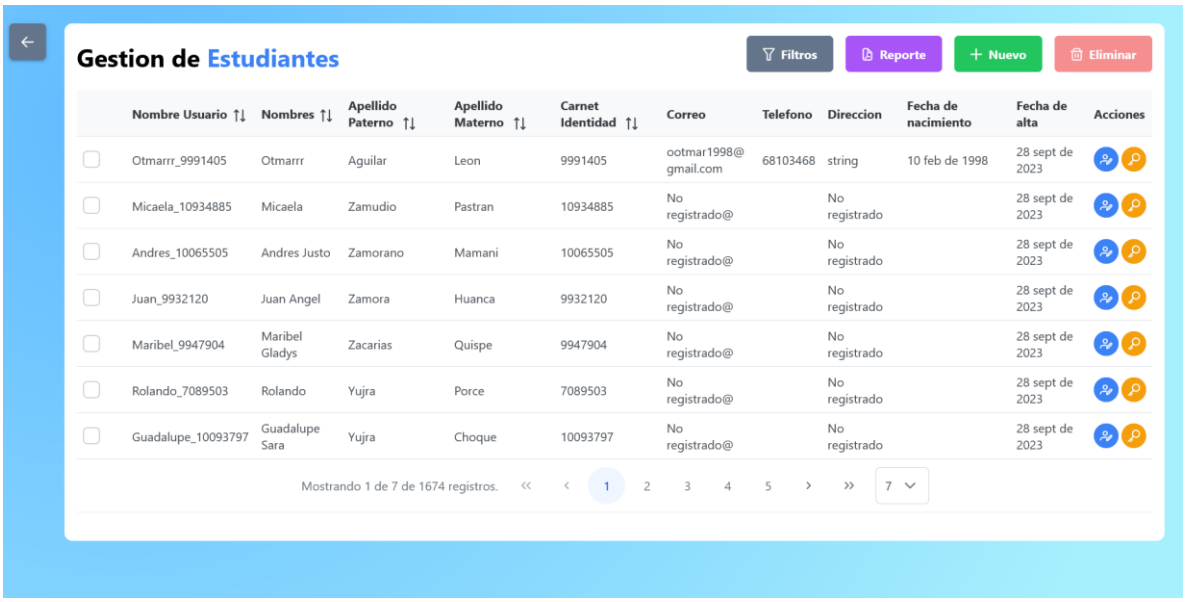
Luego click en ingresar

Gestión de usuarios

Dependiendo del rol de los usuarios se procede a seleccionar el modulo de docente o estudiante



Una vez seleccionado el modulo accedemos a la siguiente vista



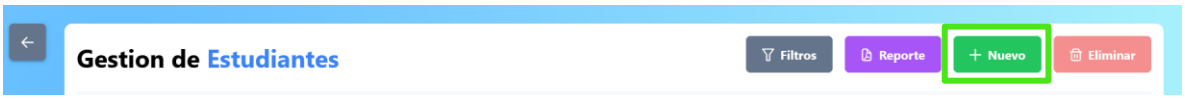
En la vista anterior se pueden realizar las siguientes acciones:

- Revisar datos de los usuarios
- Filtrar usuarios
- Genera reporte
- Crear Nuevos usuarios

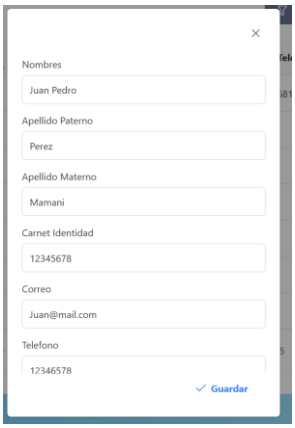
- Editar usuarios
- Eliminar usuarios
- Reiniciar contraseñas

Crear nuevo usuario

Ingresamos al botón nuevo



Aparecerá el siguiente formulario

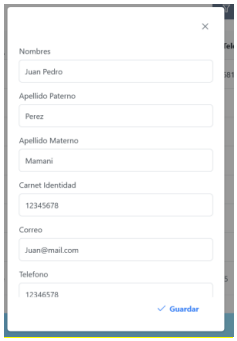


en el formulario se ingresan los datos personales del nuevo usuario y se procede a guardar

Editar usuario

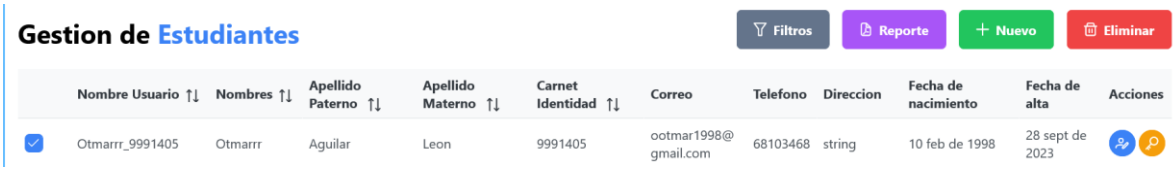
Para la edición de datos del usuario se buscar al usuario mediante los filtros y se procede a pulsar el botón siguiente botón

aparecerá el formulario con los datos del usuario para poder hacer los cambios y correcciones necesarios.



Eliminar usuario

Se puede eliminar uno o varios usuarios los cuales se han de haber seleccionado previamente con la palomilla de lado izquierdo del usuario y se habilita la opción de eliminar

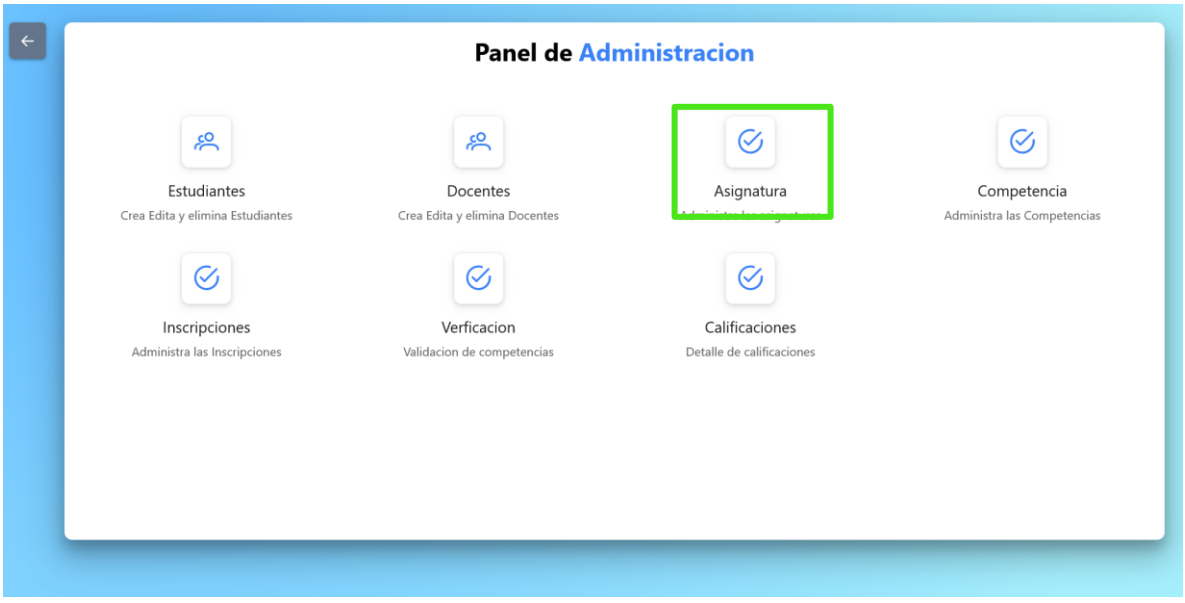


Gestión de Asignaturas

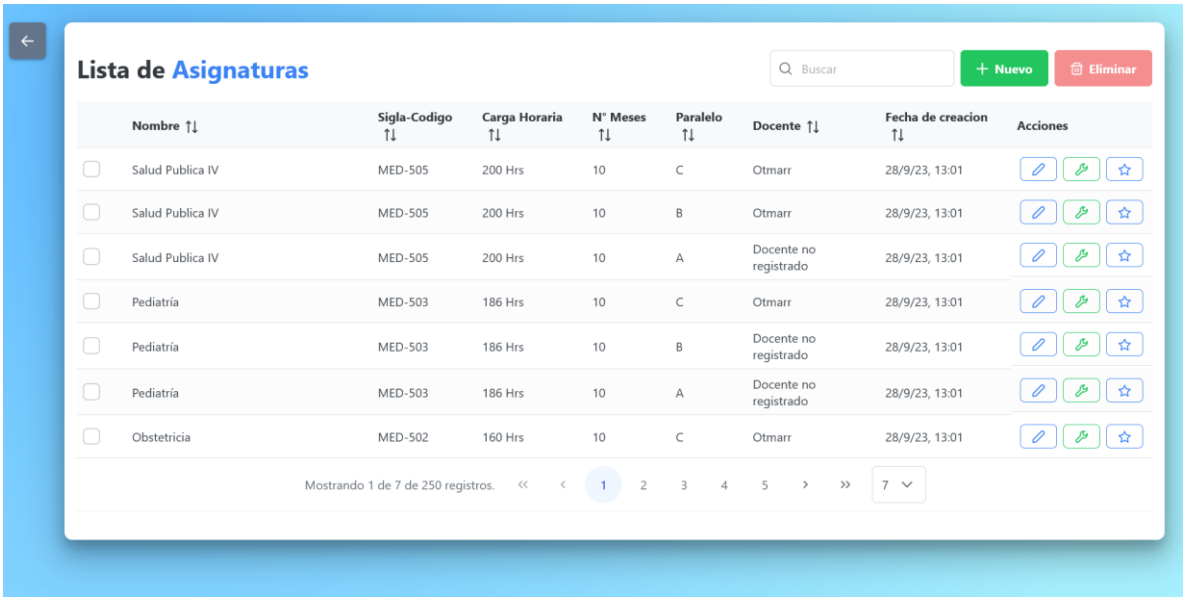
En el módulo de gestión de asignaturas se puede realizar las siguientes acciones:

- Crear Asignatura
- Editar Asignatura
- Eliminar Asignatura
- Listar Asignaturas
- Filtrar y Buscar Asignaturas
- Asignar Competencias
- Ver Calificaciones de Asignatura
- Generar Reporte de calificaciones

Para acceder al modulo de asignaturas ubicados en la vista de administrador pulsamos el botón asignaturas



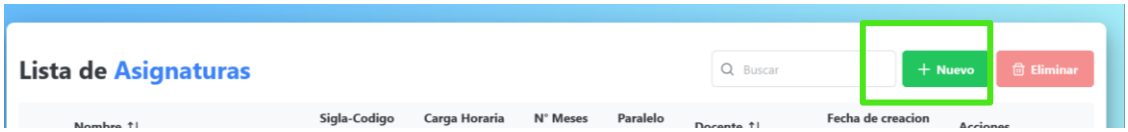
Y será redirigido a la siguiente vista



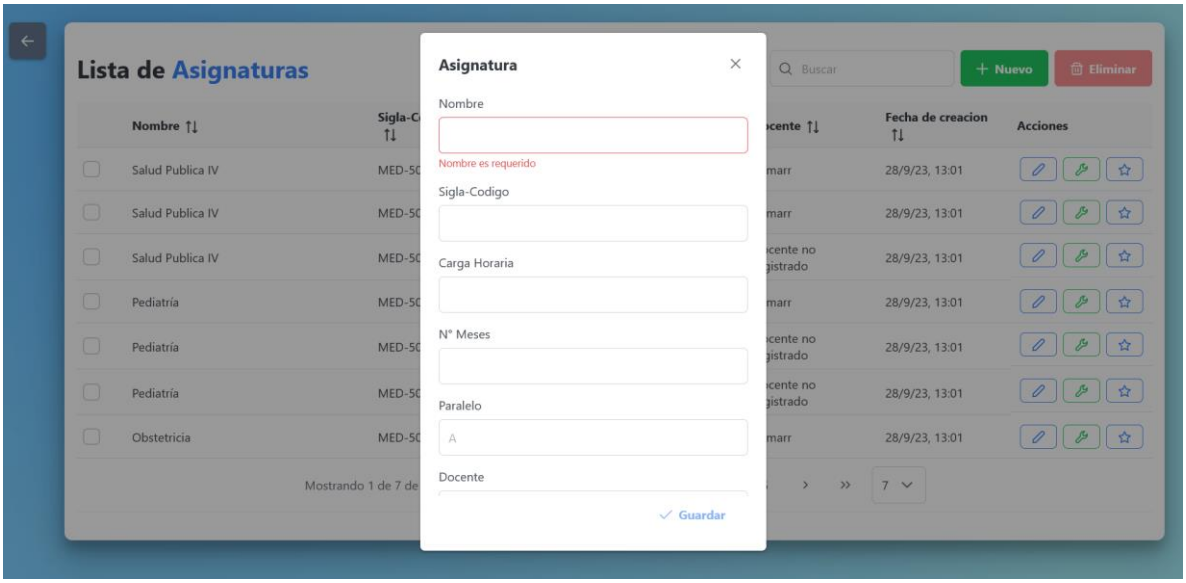
Crear Asignatura

Para la creación de una nueva asignatura se debe tener datos de dicha asignatura

Pulsamos el botón nuevo



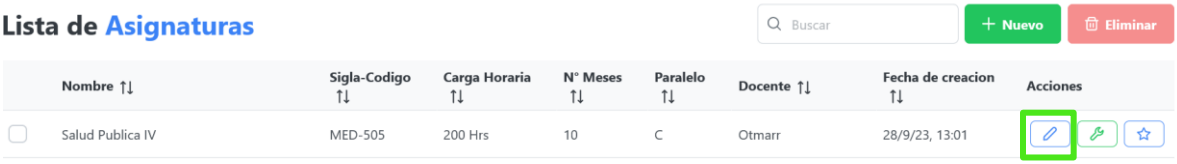
Llenamos los datos en los campos



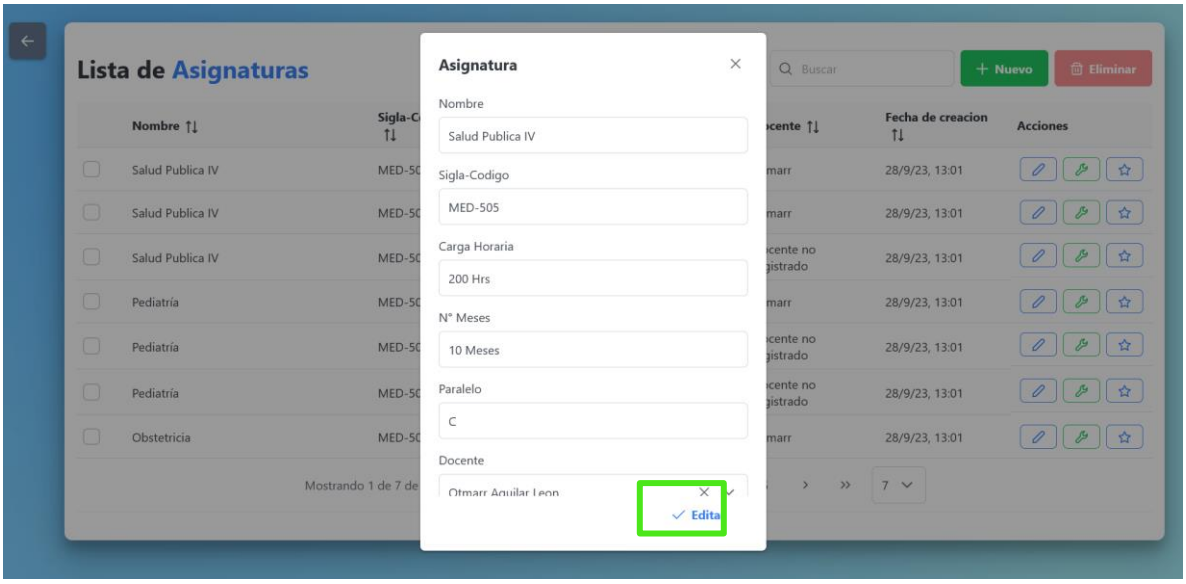
Pulsamos guardar y la asignatura aparecerá en la lista

Editar Asignatura

Buscamos la asignatura que requiere cambios y pulsamos sobre el siguiente botón:



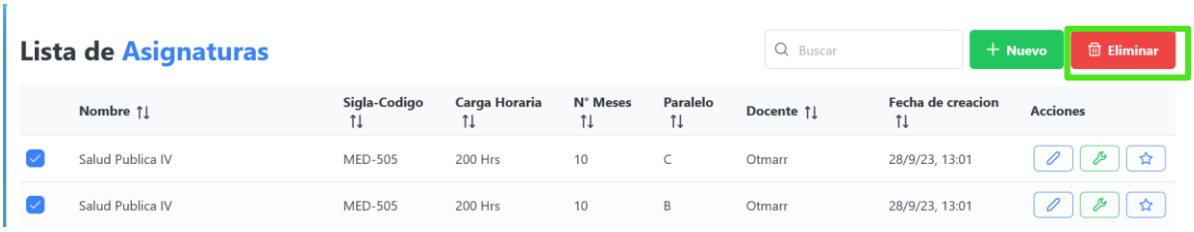
Realizamos los cambios necesarios en el siguiente formulario



Y pulsamos Editar

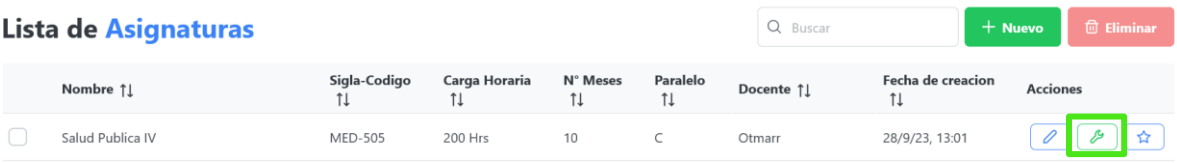
Eliminar Asignatura

Para realizar la eliminación de una o varias asignaturas seleccionamos la asignatura que se desee eliminar y pulsamos eliminar

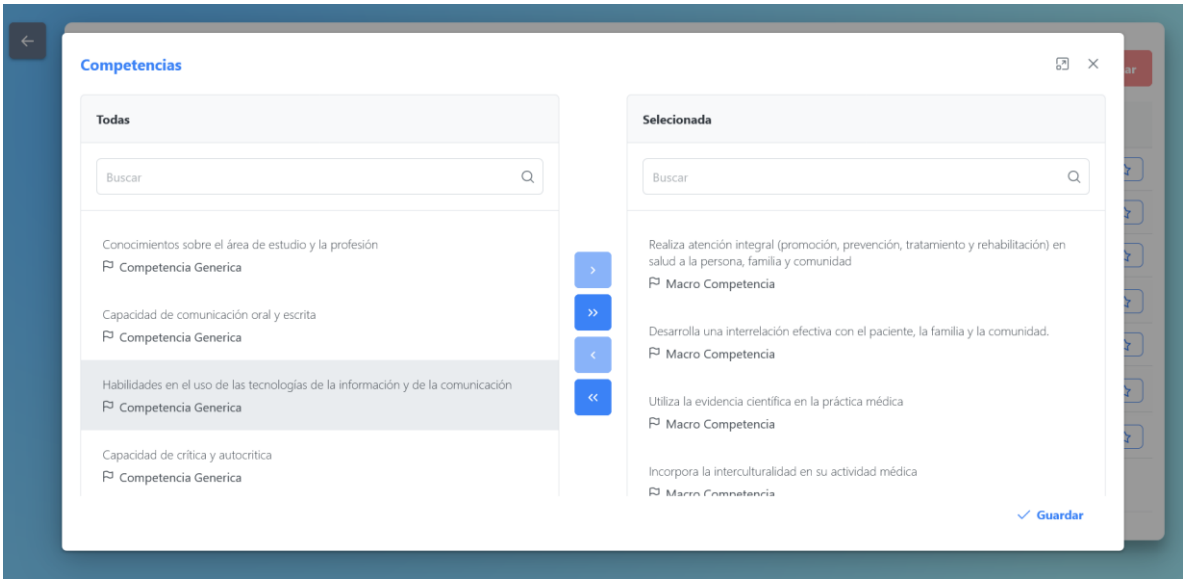


Asignación de competencias

Se puede asignaturas las competencias a la asignatura una vez dichas competencias ya estén creadas, para asignar la competencia pulsamos sobre el siguiente botón:

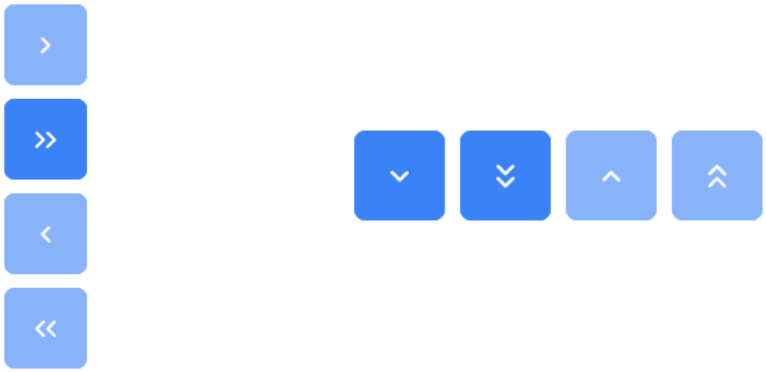


Se desplegará la siguiente vista



Las competencias listadas en la izquierda son todas las competencias ya creadas, las competencias de la derecha son competencias ya asignadas a la asignatura.

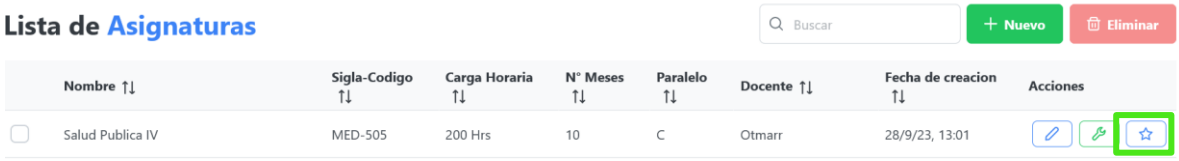
Para asignar las competencias se realiza mediante los siguientes controles Las flechas de en medio pueden llevar las competencias de derecha a izquierda y viceversa, al pulsar sobre una o varias competencias se habilita la flecha simple la cual llevara la competencia a la izquierda, las flechas dobles sirven para llevar todas las competencias.



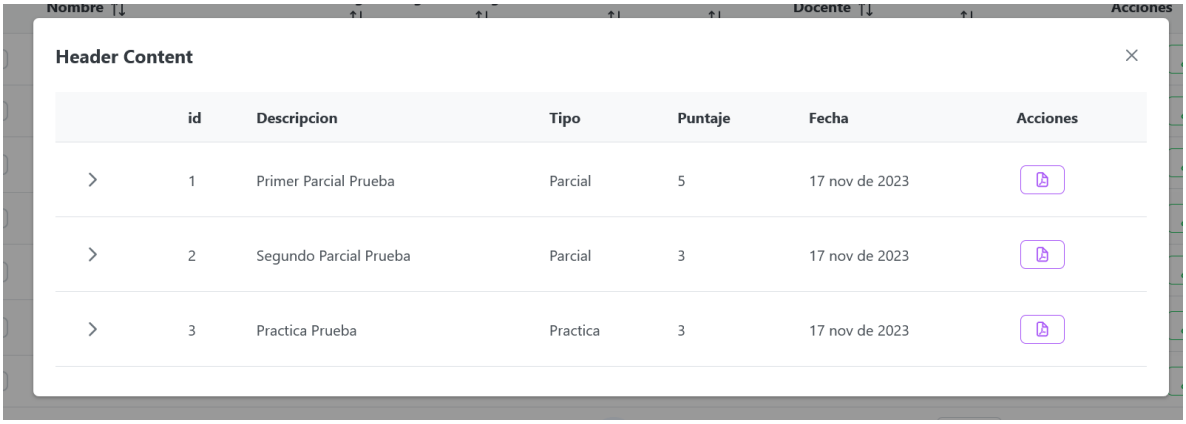
Por ultimo se pulsa el botón guardar.

Revisar y generar Reportes de calificación

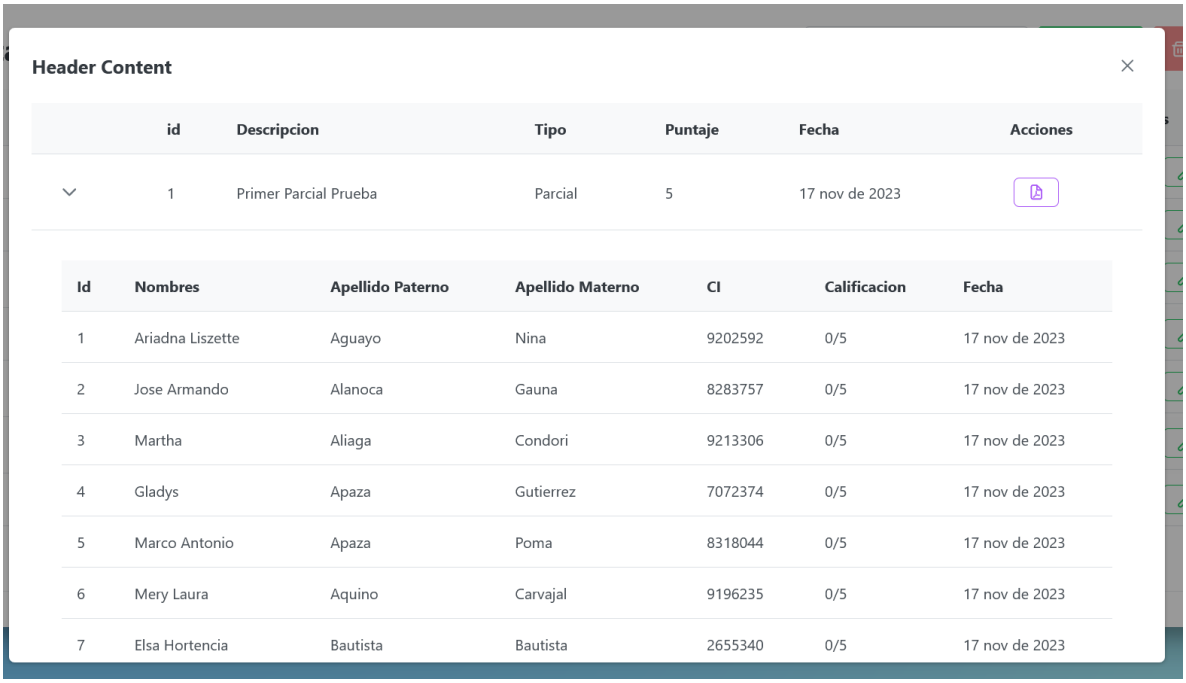
Para poder revisar las calificaciones que tiene una asignatura pulsar el siguiente botón



Se desplegará la siguiente vista



Aparece la lista de las calificaciones creadas en la asignatura, se puede generar un reporte de las calificaciones o ver las calificaciones de los estudiantes



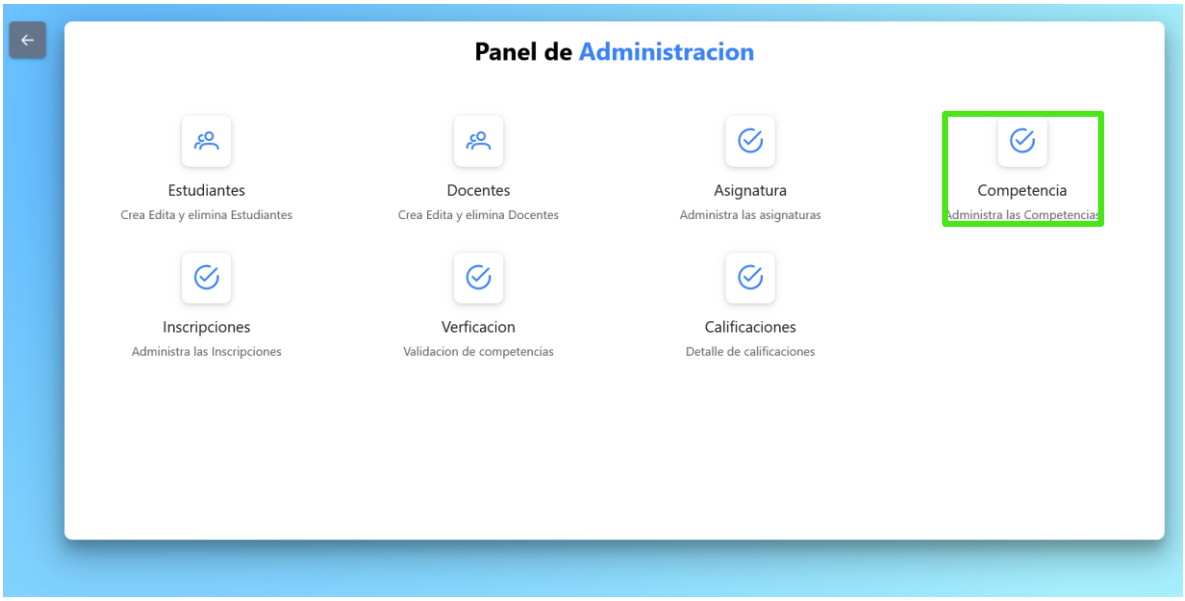
id	Nombres	Apellido Paterno	Apellido Materno	C.I.	Calificacion	Fecha
1	Ariadna Liszette	Aguayo	Nina	9202592	0	17/11/2023
2	Jose Armando	Alanoca	Gauna	8283757	0	17/11/2023
3	Martha	Aliaga	Condori	9213306	0	17/11/2023
4	Gladys	Apaza	Gutierrez	7072374	0	17/11/2023
5	Marco Antonio	Apaza	Poma	8318044	0	17/11/2023
6	Mery Laura	Aquino	Carvajal	9196235	0	17/11/2023
7	Elsa Hortencia	Bautista	Bautista	2655340	0	17/11/2023
8	Raymi	Berrios	Aro	9986494	0	17/11/2023
9	Carmen Domy	Blanco	Rosales	9867385	0	17/11/2023
10	Patricia	Cala	Condori	6986340	0	17/11/2023
11	Valery Gimena	Calle	Acarapi	11089965	0	17/11/2023
12	Yhoselyn	Callisaya	Mamani	7024552	0	17/11/2023
13	Lorena	Callisaya	Quisbert	9994047	0	17/11/2023
14	Pamela	Canaviri	Lazaro	6667853	0	17/11/2023

Gestión de competencias

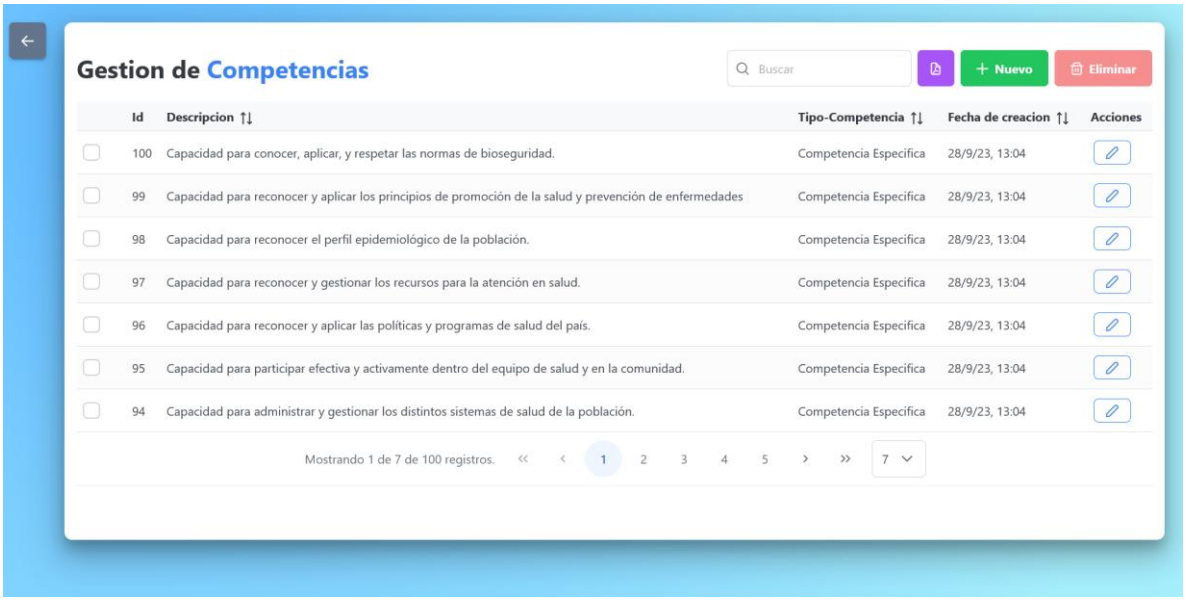
En el módulo de gestión de competencias se puede realizar las siguientes acciones:

- Crear Competencia
- Editar Competencia
- Eliminar Competencia
- Listar Competencia
- Filtrar y Buscar Competencia

Para acceder al módulo de asignaturas ubicados en la vista de administrador pulsamos el botón asignaturas

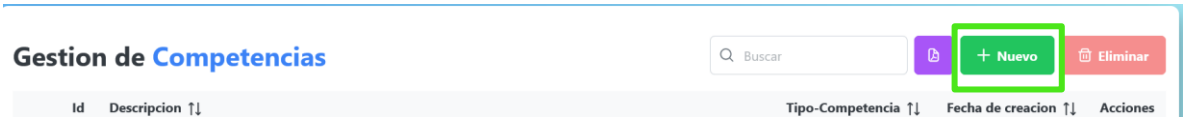


Y será redirigido a la siguiente vista

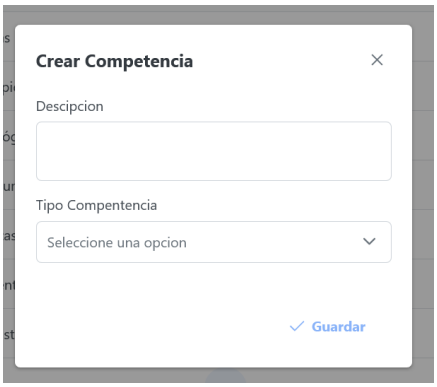


Nueva Competencia

Para la creación de una nueva competencia pulsamos el siguiente botón:



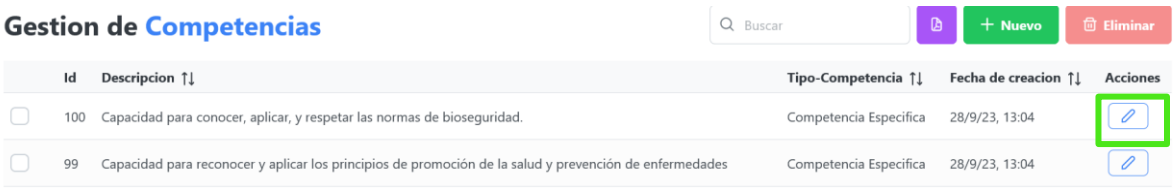
Llenar el formulario con los datos de la competencia



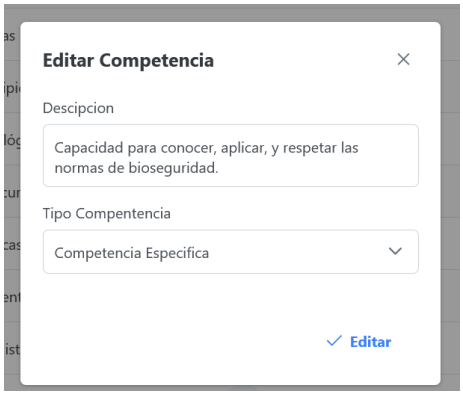
Una vez llenados los datos pulsar guardar y la competencia aparecerá en el listado

Editar Competencia

Para editar la competencias buscamos la competencia a editar y pulsamos el siguiente botón:

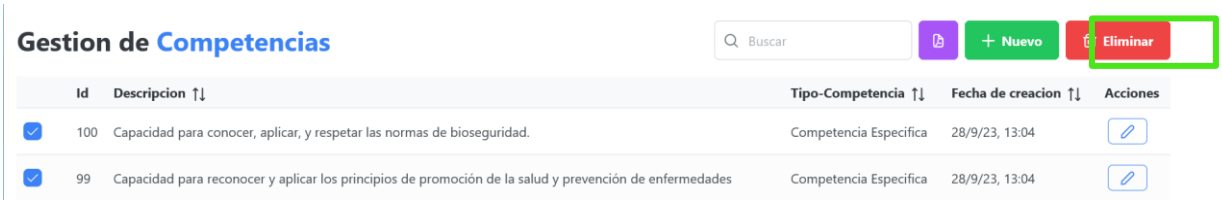


Se realiza los cambios necesarios y se pulsa guardar.



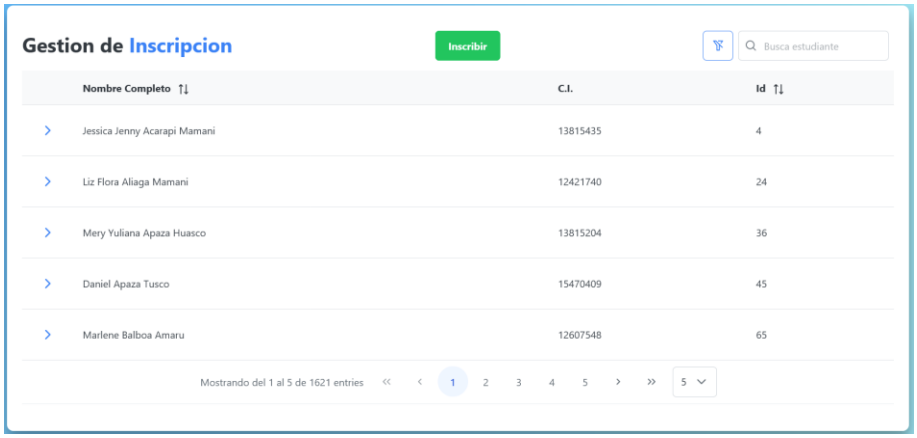
Eliminar Competencias

Para eliminar una o varias competencias se busca y selecciona la competencia a eliminar y se pulsa el botón eliminar



Gestión de Inscripción

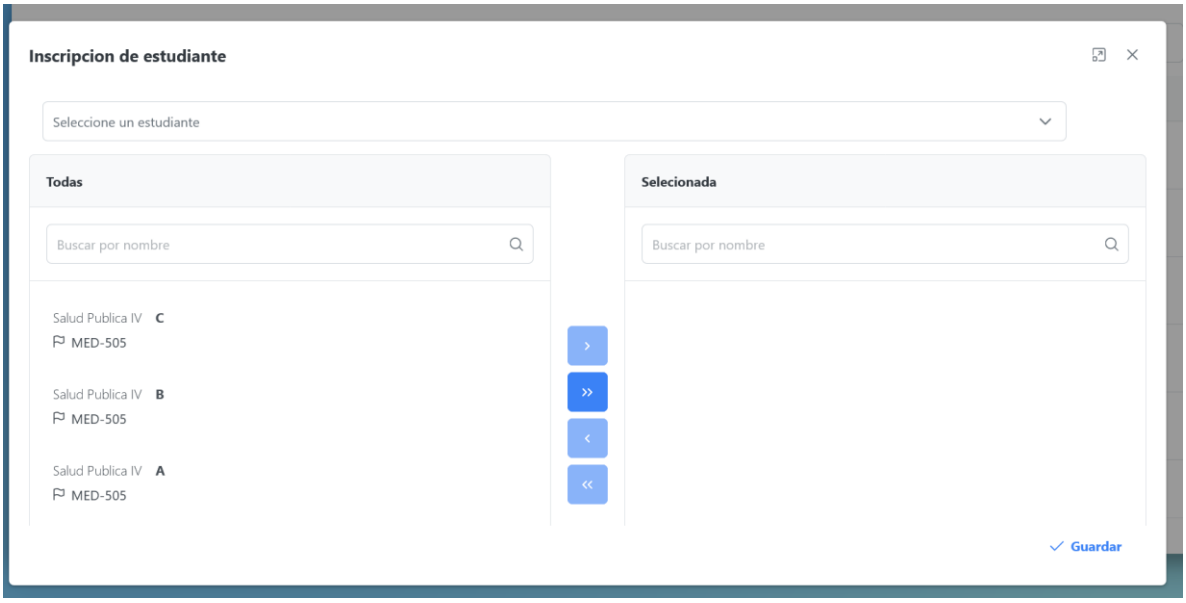
En el módulo de inscripción se muestran la lista de todos los estudiantes inscritos en una o varias asignaturas se pueden buscar y filtrar los estudiantes



Inscripción

Pulsamos el botón inscribir

Se desplegará la siguiente vista:



En la parte superior se muestra la lista de todos los estudiantes registrados, en la izquierda todas asignaturas registradas, Para realizar la inscripción seleccionar un estudiante y llevar las asignaturas en la cuales queremos inscribir a la derecha.

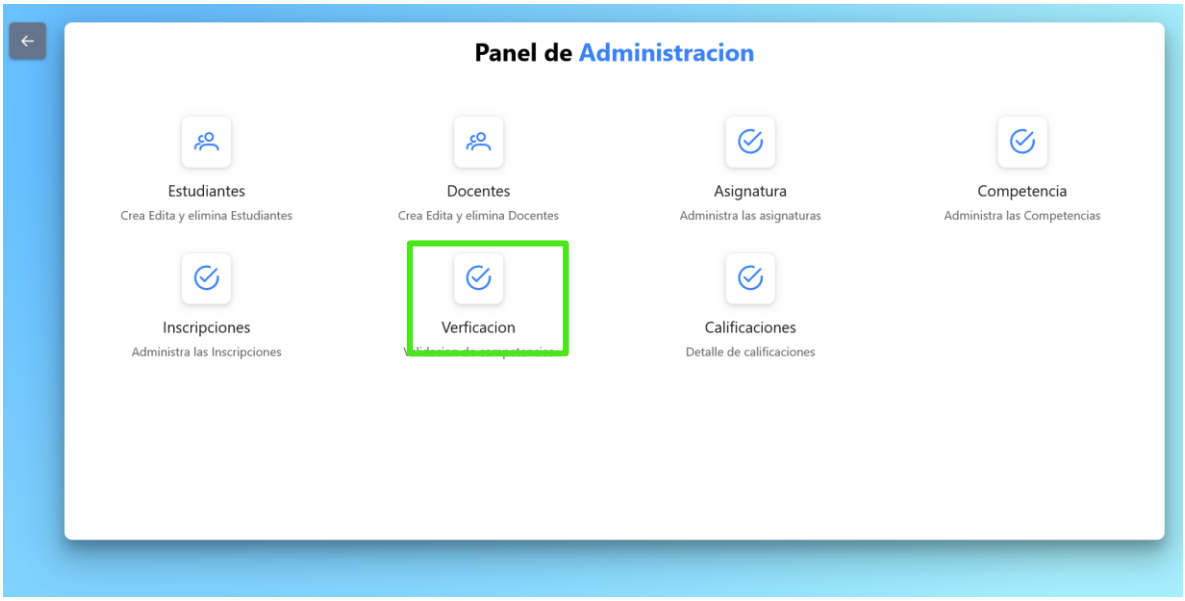
Pulsamos guardar y el estudiante será inscrito en las asignaturas seleccionadas.

Módulo de verificación

En el módulo verificación se puede realizar las siguientes acciones:

- Listado de competencias por estudiantes
- Filtrar y buscar estudiantes por competencias
- Generar reportes en base a las competencias
- Verificación perfil profesional del estudiante

Para acceder al módulo de asignaturas ubicados en la vista de administrador pulsamos el botón Verificación



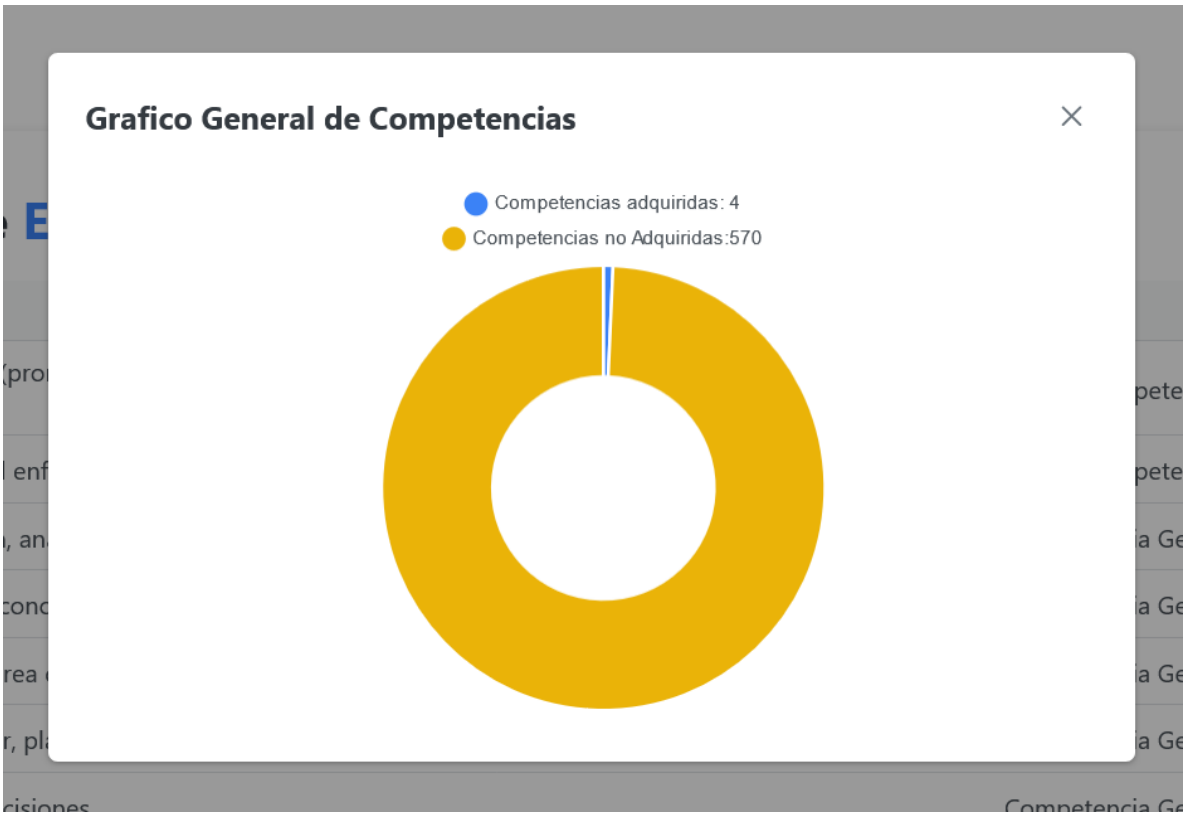
Y será redirigido a la siguiente vista

Verificación de Perfil Profesional

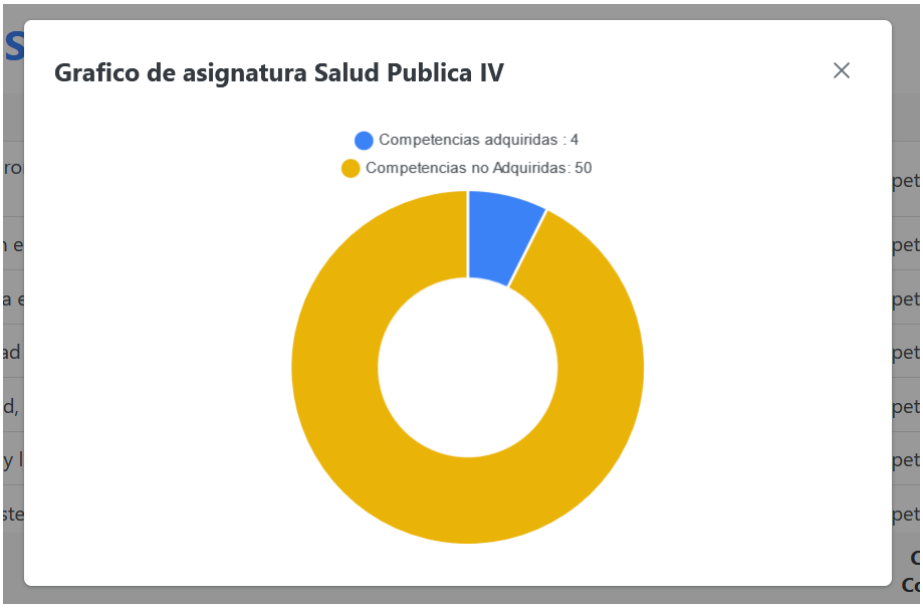
Para acceder a la verificación del perfil profesional del estudiante pulsamos el botón:

Se mostrará la siguiente vista en cual se lista por asignatura las competencias obtenidas y no obtenidas

Al pulsar el botón de Grafico General nos mostrara el conteo general de todas las competencias no obtenidas y obtenidas por el estudiante



Al pulsar botón grafico nos mostrara por asignatura las competencias obtenidas por el estudiante



Salir

Al pulsar sobre el botón Salir se cerrará la sesión y lo redireccionará al login