



UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS
FACULTAD 4

**SISTEMA INFORMÁTICO PARA LA GESTIÓN DE LA INFORMACIÓN DE LOS
TRABAJOS DE DIPLOMAS DE LA UNIVERSIDAD DE LAS CIENCIAS
INFORMÁTICAS**

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: Abel Machado Montes

Tutor: Ing. Jorge Emilio Badía Valdés

La Habana, 2023

*"Nuestra recompensa se encuentra en el esfuerzo y no en el resultado.
Un esfuerzo total es una victoria completa."
Mahatma Gandhi*

Dedicatoria

A mi familia y amigos, que me apoyaron en todo momento.

Agradecimientos

A todos los que de una forma u otra contribuyeron a la realización de este trabajo

Declaración de autoría

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales sobre esta, con carácter exclusivo.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

Abel Machado Montes
Autor

Ing. Jorge Emilio Badía Valdés
Tutor

El uso de las Tecnologías de la Información y las Comunicaciones (TIC) en la sociedad produce cambios trascendentales en todas las esferas de la vida. Cada vez son más las organizaciones e instituciones que se inclinan por incorporar aplicaciones que gestionen su información, obteniendo así una mayor dinámica en sus procesos. La Universidad de las Ciencias Informáticas (UCI) no cuenta actualmente con un sistema que permita la gestión de todo el proceso que se lleva a cabo en los años terminales con los trabajos de diploma. Por lo cual, el seguimiento al ciclo de vida de estos se vuelve un proceso largo y tedioso, donde la gran cantidad de información generada se almacena en archivos por separados, lo que crea una descentralización de la información. Además, no se posee un control sobre los temas destinados para dichos trabajos, por lo que solo pueden ser asignados en los centros y/o departamentos de la facultad. La creación de los tribunales se realiza mediante el uso de una hoja de cálculo y solo se encuentra disponible de forma local en las computadoras de los implicados en el proceso. La presente investigación tiene como objetivo desarrollar un sistema informático que contribuya a la gestión de la información asociada a los trabajos de diploma de la UCI. Para guiar el proceso de desarrollo de la propuesta se utilizó como metodología de software XP, como framework de desarrollo Django, como lenguaje de desarrollo se utilizó Python. Para evaluar la calidad de la solución se realizaron pruebas de software donde se pudo corroborar el cumplimiento de los requisitos acordados con el cliente.

Palabras clave: gestión, informatización, tesis, trabajos de diploma.

Introducción	1
1 Marco teórico referencial sobre el sistema para la gestión de los trabajos de diploma en la Facultad 4 de la Universidad de las Ciencias Informáticas (UCI)	5
1.1 Análisis sobre la gestión de los trabajos de diploma	5
1.1.1 Tesis de grado o Trabajo de diploma	5
1.1.2 Gestión de información	7
1.2 Análisis de las soluciones existentes	7
1.2.1 Ámbito internacional	8
1.2.2 Ámbito nacional	9
1.2.3 Consideraciones del estudio de los sistemas	11
1.3 Análisis de las metodologías, herramientas y tecnologías para el desarrollo del sistema	11
1.3.1 Metodología	11
1.3.2 Herramientas y tecnologías	13
1.4 Conclusiones parciales	17
2 Características y diseño del sistema para la gestión de los trabajos de diploma en la Facultad 4 de la UCI	19
2.1 Descripción de la propuesta de solución	19
2.2 Especificación de requisitos del sistema	20
2.2.1 Requisitos funcionales	20
2.2.2 Requisitos no funcionales	21
2.3 Descripción de las historias de usuario	22
2.4 Planificación de la entrega	26
2.4.1 Estimación del esfuerzo por Historias de Usuario	26
2.5 Plan de iteraciones	28
2.5.1 Desarrollo del plan de iteraciones	28
2.6 Tarjetas Clase-Responsabilidad-Colaboración (CRC)	30
2.7 Estilo arquitectónico	31
2.7.1 Patrón arquitectónico	31

2.8	Patrones de diseño	32
2.9	Modelo de datos	34
2.10	Consideraciones parciales	35
3	Implementación y pruebas del sistema para la gestión de los trabajos de diploma en la	
	Facultad 4 de la UCI	36
3.1	Estándares de codificación	36
3.2	Diagrama de despliegue	39
3.3	Tareas de ingeniería	40
3.3.1	Tareas de ingeniería agrupadas por iteración	40
3.4	Pruebas de software	43
3.4.1	Pruebas de aceptación	43
3.4.2	Prueba de seguridad	46
3.4.3	Pruebas unitarias	47
3.4.4	Pruebas funcionales	49
3.4.5	Prueba de rendimiento	52
3.5	Consideraciones parciales	53
	Conclusiones	55
	Recomendaciones	56
	Referencias bibliográficas	57
	Apéndices	61
.1	Modelo de entrevista aplicada a profesores	62
.2	Guía de observación para el análisis de los sistemas similares	62
.3	Historias de usuario	62
.4	Casos de pruebas de Aceptación	83
.5	Casos de prueba para las pruebas unitarias	92

Índice de figuras

1.1	Interfaz del sistema SIDEP	8
1.2	Interfaz del sistema Teseo	9
1.3	Interfaz del sistema SIGENU	10
1.4	Interfaz del sistema Akademos	10
1.5	Interfaz del sistema REPXOS 4.0	11
2.1	Representación de la arquitectura.	32
2.2	Modelo de datos	35
3.1	Representación de la arquitectura. (Fuente: Elaboración propia)	39
3.2	Prueba de seguridad. (Fuente: Elaboración propia)	46
3.3	Prueba de seguridad. (Fuente: Elaboración propia)	47
3.4	Representación del resultado de las pruebas. (Fuente: Elaboración propia)	52
3.5	Resultado de pruebas de rendimiento	53
3.6	Resultado de pruebas de rendimiento	53

Índice de tablas

1.1	Diferencias entre metodologías ágiles y tradicionales (Fuente: (Sangama Oñate, 2020)). . . .	12
2.1	Descripción de los Requisitos Funcionales. (Fuente: Elaboración propia).	20
2.2	Historia de usuario # 1	22
2.3	Historia de usuario # 2	23
2.4	Historia de usuario # 3	24
2.5	Historia de usuario # 4	25
2.6	Historia de usuario # 5	26
2.7	Estimación de esfuerzo por Historias de Usuario (HU). (Fuente: Elaboración propia). . . .	27
2.8	Pan de iteraciones (Fuente: Elaboración propia).	28
2.9	Tarjeta CRC # 1	30
2.10	Tarjeta CRC # 2	30
2.11	Tarjeta CRC # 3	31
3.1	Estándares de codificación a utilizar en la implementación del sistema. (Fuente: Elaboración propia).	36
3.2	Historias de usuario abordadas en la primera iteración (Fuente: Elaboración propia).	40
3.3	Historias de usuario abordadas en la segunda iteración (Fuente: Elaboración propia).	40
3.4	Historias de usuario abordadas en la tercera iteración (Fuente: Elaboración propia).	41
3.5	Tarea de ingeniería # 1	42
3.6	Tarea de ingeniería # 2	42
3.7	Tarea de ingeniería # 3	42
3.8	Tarea de ingeniería # 4	42
3.9	Tarea de ingeniería # 5	43
3.10	Prueba de aceptación # 1	44
3.11	Prueba de aceptación # 2	44
3.12	Prueba de aceptación # 3	44
3.13	Prueba de aceptación # 4	45
3.14	Prueba de aceptación # 5	45
3.15	Prueba de aceptación # 6	45
3.16	Cálculo de la complejidad ciclomática del método crear_Tribunal().	47

3.17	Caminos del grafo de flujo (Fuente: Elaboración propia).	48
3.18	Caso de prueba “Autenticar usuario” (Fuente: Elaboración Propia).	49
3.19	Variables de caso de prueba “Autenticar usuario” (Fuente: Elaboración Propia).	50
3.20	Caso de prueba “Registrar usuario” (Fuente: Elaboración Propia).	50
3.21	Variables de caso de prueba “Registrar usuario” (Fuente: Elaboración Propia).	51
22	Guía de observación para estudio de soluciones (Fuente: Elaboración propia).	62
23	Historia de usuario # 6	62
24	Historia de usuario # 7	63
25	Historia de usuario # 8	64
26	Historia de usuario # 9	64
27	Historia de usuario # 10	65
28	Historia de usuario # 11	65
29	Historia de usuario # 12	66
30	Historia de usuario # 13	67
31	Historia de usuario # 14	67
32	Historia de usuario # 15	68
33	Historia de usuario # 16	69
34	Historia de usuario # 17	70
35	Historia de usuario # 18	70
36	Historia de usuario # 19	71
37	Historia de usuario # 20	72
38	Historia de usuario # 21	72
39	Historia de usuario # 22	73
40	Historia de usuario # 23	73
41	Historia de usuario # 24	74
42	Historia de usuario # 25	75
43	Historia de usuario # 26	76
44	Historia de usuario # 27	77
45	Historia de usuario # 28	78
46	Historia de usuario # 29	79
47	Historia de usuario # 30	79
48	Historia de usuario # 31	80
49	Historia de usuario # 32	81
50	Historia de usuario # 33	81
51	Historia de usuario # 34	82
52	Historia de usuario # 35	83
53	Prueba de aceptación # 7	83

54	Prueba de aceptación # 8	84
55	Prueba de aceptación # 9	85
56	Prueba de aceptación # 10	85
57	Prueba de aceptación # 11	86
58	Prueba de aceptación # 12	86
59	Prueba de aceptación # 13	87
60	Prueba de aceptación # 14	87
61	Prueba de aceptación # 15	88
62	Prueba de aceptación # 16	88
63	Prueba de aceptación # 17	88
64	Prueba de aceptación # 18	89
65	Prueba de aceptación # 19	89
66	Prueba de aceptación # 20	90
67	Prueba de aceptación # 21	90
68	Prueba de aceptación # 22	91
69	Prueba de aceptación # 23	91
70	Caso de Prueba para el camino básico 1 (Fuente: Elaboración propia).	92

En un mundo tan globalizado y lleno de nuevas tecnologías y concepciones sociales, políticas, económicas y filosóficas como el actual, no se puede negar la importancia que tiene la educación, y en específico el papel de las Instituciones de Educación Superior (IES). Según un documento de la Liga de Universidades Europeas de Investigación (LERU, por sus siglas en inglés), las IES han aumentado su importancia y sus funciones en la sociedad mundial (Boulton y Lucas, 2011). Siempre fueron fuente de pensamiento radical y progreso social y se han convertido en uno de los principales elementos que aportan conocimiento científico a la sociedad.

Las IES han tomado el liderazgo en cuanto al desarrollo de concepciones multidisciplinarias que son vitales cuando se habla de muchos de los desafíos a los que se enfrenta el mundo. Son centros que han ganado en cuanto a flexibilidad y adaptabilidad, y que se encuentran en constante cambio y comprometidos en la búsqueda y expansión del conocimiento, así como el tratamiento de las necesidades de la sociedad. Son las encargadas de moldear a sus estudiantes inculcándoles principios de razonamiento y acción, conforme a las necesidades actuales. Prueba de todo esto, lo ha sido la atención y la importancia que les han dado los gobiernos y sociedades, que ven en ellas una fuente de conocimiento y pensamiento innovador, que provee personal altamente calificado y que contribuye al mantenimiento de la sociedad.

En Cuba, el Ministerio de Educación Superior (MES) es el organismo encargado de dirigir, ejecutar y controlar la política estatal y del gobierno sobre las IES, sustentado en una concepción humanista, científica, tecnológica, innovadora, integrada y comprometida con la sociedad cubana para construir un socialismo próspero y sostenible (MES, 2022). Una de las IES pertenecientes al MES es la UCI, universidad fundada en el año 2002, por iniciativa del comandante en jefe Fidel Castro. Este centro tiene como misión principal la formación de profesionales comprometidos con la patria y estén altamente calificados en las ramas de las tecnologías. Por lo que la UCI constituye para el país un eslabón clave para el proceso de informatización de la sociedad y a la transformación digital, ambos aspectos son tratados hoy en el Plan nacional de desarrollo económico social hasta el 2030 (Cuba (PCC), 2016) que propone el impulso a las actividades de ciencia, tecnología e innovación y la transformación digital.

La UCI se caracteriza por ser una universidad de nuevo tipo, donde uno de sus procesos sustantivos es la producción, por lo que su modelo de formación está basado en el principio estudio - trabajo, mediante la incorporación de los estudiantes a proyectos reales de investigación y desarrollo. Por lo que el currículo del Ingeniero en Ciencias Informáticas se divide en dos grandes ciclos, denominados: ciclo de integración básico y ciclo de integración profesional. En el ciclo de formación básica, el énfasis se hace en la formación

académica y tiene como objetivo fundamental la preparación para la incorporación al ciclo profesional. Por otro lado, en el ciclo de integración profesional el peso fundamental lo tiene la formación desde lo laboral y su objetivo fundamental es concretar el desempeño de los estudiantes en situaciones profesionales reales.

El MES en la Resolución 47 del 2022 referente al Reglamento para el trabajo docente metodológico en la Educación Superior en el Capítulo XII, artículo 321.1 referente a la evaluación de la culminación de estudios plantea que existen dos tipos de evaluaciones, uno los exámenes estatales y la defensa del trabajo de diploma (Educación Superior, 2022). En el caso de la UCI los estudiantes de años terminales son evaluados con la defensa de trabajo de diploma. Según la resolución este ejercicio tiene como objetivo comprobar el grado de dominio de los estudiantes de los objetivos generales de la carrera, mediante la solución, con independencia y creatividad, de un problema propio de la profesión, utilizando la metodología de investigación científica. El ejercicio final de culminación de estudios para la carrera, es un Trabajo de Diploma que está concebido, como un trabajo de ingeniería en alguno de los perfiles del ejercicio de la profesión del ingeniero informático. Este ejercicio es el resultado de un proceso, que transita por varias fases, las cuales se describen a continuación:

- Elaboración del perfil de tesis: en esta fase los aspirantes y tutores participan en la elaboración de los perfiles de tesis siguiendo las pautas establecidas en cada Facultad/Área.
- Constitución del Comité o Comisión de Tesis (CT): se conforma en las facultades con la aprobación del Decano(a) y es el órgano docente encargado de la toma de decisiones sobre el proceso de desarrollo y seguimiento de los trabajos de diploma. Este órgano es el encargado de la clasificación de los proyectos de tesis en correspondencia con su área del conocimiento, donde se genera un acta de constitución de los CT.
- Revisión y aprobación de los perfiles de tesis: en esta fase juegan un papel activo los miembros del CT y los tutores. Una vez aprobados los perfiles, no podrán modificarse el objetivo general o el problema de la investigación científica definidos inicialmente en el perfil del Trabajo de Diploma, exceptuando los casos avalados por el CT, de esta fase se genera un acta de aprobación, siguiendo las pautas establecidas en la Facultad/Universidad.
- Realización de los seminarios de tesis: este es un acto de carácter evaluativo dentro del desarrollo del Trabajo de Diploma. En cada seminario debe dejar constancia escrita de las valoraciones realizadas por los miembros del CT, el tutor y el aspirante. De igual manera, se debe reportar al Vicedecano(a) de Formación un resumen del estado de los trabajos de diploma, incidencias y acciones correctivas en caso de ser necesario.
- Constitución de los Tribunales de Tesis: los tribunales y oponentes de los trabajos de diploma son definidos por el CT y aprobados por el Decano(a) de la Facultad, donde se genera una resolución decanal, siguiendo las pautas establecidas en la Facultad/Universidad.
- Defensa de trabajos de diploma: es un ejercicio académico donde se evalúa la calidad de la investigación desarrollada, el documento de tesis entregado, la defensa que se realiza en acto público ante el tribunal, y la expresión de un comportamiento ético y moral conforme a la culminación de estudios

de un Ingeniero en Ciencias Informáticas.

El seguimiento del ciclo de vida de los trabajos de diplomas es un proceso largo y tedioso. La gran cantidad de información generada se almacena en archivos por separados lo que crea una descentralización de la información; lo que puede llevar a la pérdida o desinformación. Además, no se posee un control sobre los temas destinados para dichos trabajos, por lo que solo pueden ser asignados en los centros y/o departamentos de la facultad, lo que genera atraso en otros locales que no poseen temas propios para ser asignados. Incluido a lo anterior está la creación de los tribunales; proceso que se realiza mediante el uso de una hoja de cálculo y solo se encuentra disponible de forma local en las computadoras de los implicados en el proceso, lo que evita la disponibilidad de la información para personas no involucradas directamente. A partir de la problemática antes descrita se identifica como **problema de investigación**: ¿Cómo contribuir con la gestión de la información de los trabajos de diplomas en la Facultad 4 de la Universidad de las Ciencias Informáticas?

A partir del problema de investigación formulado, se define como **objeto de estudio**: la gestión de la información del proceso de trabajo de diploma, y dentro de este se precisa como **campo de acción**: la gestión de información asociada a los trabajos de diploma de la Facultad 4 en la Universidad de las Ciencias Informáticas. Para darle solución al problema antes planteado, se define como **objetivo general**: desarrollar un sistema informático que contribuya a la gestión de la información asociada a los trabajos de diploma en la Facultad 4 de la Universidad de las Ciencias Informáticas.

Del mismo se derivan los siguientes **objetivos específicos**:

1. Elaboración del marco teórico de la investigación a partir del estado del arte existente sobre la gestión de los trabajos de diploma.
2. Analizar el estado actual de la gestión de la información asociada a los trabajos de diploma en la Facultad 4 de la UCI.
3. Diseñar el sistema informático para la gestión de información asociada a los trabajos de diploma en la Facultad 4 de la UCI
4. Implementar el sistema informático para la gestión de información asociada a los trabajos de diploma en la Facultad 4 de la UCI
5. Validar el sistema informático para la gestión de información asociada a los trabajos de diploma en la Facultad 4 de la UCI

Para darle solución a los objetivos trazados en la investigación se aplicaron los siguientes **Métodos Científicos**: **Métodos teóricos**

- Histórico-Lógico: se emplea para identificar posibles funcionalidades que pueda tener el sistema a partir del análisis de la evolución de los sistemas de gestión de trabajos de diploma.
- Análisis-Síntesis: empleado para el análisis, evaluación y selección de las técnicas a emplear en el desarrollo del sistema. Así como para sintetizar la información que se obtuvo mediante la entrevista

con el cliente de manera que pudiera ser usada en el desarrollo del mismo, además, en la identificación de los elementos del marco teórico de la investigación.

- **Modelación:** para realizar una representación del proceso estudiado que sirva de guía en el desarrollo del sistema, y mediante este, identificar las características y relaciones fundamentales.

Métodos empíricos

- **Entrevista:** empleado en los encuentros con el cliente para obtener la información necesaria que permita determinar las características, cualidades y requisitos con los que debe contar la propuesta de solución.
- **Observación:** empleado para obtener el conocimiento necesario del funcionamiento de las soluciones existentes para la gestión del proceso de los trabajos de diploma.

El presente trabajo de diploma está estructurado de la siguiente manera: introducción, tres capítulos, conclusiones generales, recomendaciones y las referencias bibliográficas empleadas durante el desarrollo de la investigación, y por último para complementar la investigación se presentan una serie de anexos.

A continuación, se muestra la descripción de los capítulos:

- **Capítulo 1. Marco teórico referencial sobre el sistema para la gestión de los trabajos de diploma de la UCI:** se realiza un estudio del estado del arte sobre los principales aspectos de la investigación. Se describen el proceso de desarrollo de software, así como las tendencias, técnicas, metodología y tecnologías usadas en la propuesta de solución.
- **Capítulo 2. Características y diseño del sistema para la gestión de los trabajos de diploma de la UCI:** se identifican y describen los conceptos asociados al dominio del problema y los procesos relacionados con el negocio teniendo en cuenta la metodología seleccionada en el capítulo anterior. Se definen cuáles son los requerimientos funcionales, no funcionales y el modelo de datos.
- **Capítulo 3. Implementación y pruebas del sistema para la gestión de los trabajos de diploma de la UCI:** en este capítulo se define el estándar de codificación que sirve de guía para la implementación de la solución propuesta, así como la estrategia de pruebas a aplicar para lograr un correcto funcionamiento.

Marco teórico referencial sobre el sistema para la gestión de los trabajos de diploma en la Facultad 4 de la UCI

En este capítulo se definen los principales conceptos que sustentan el dominio de la investigación y que posibilitan un entendimiento profundo del tema. Se expone una valoración del estado del arte de la investigación realizada con el estudio de algunas soluciones parecidas y tendencias actuales a la hora de resolver problemas similares. Se describen, además, las principales tecnologías, herramientas y la metodología que se pretende usar para el desarrollo del sistema.

1.1. Análisis sobre la gestión de los trabajos de diploma

1.1.1. Tesis de grado o Trabajo de diploma

La palabra “tesis” no procede del campo de la metodología de la investigación ni de la propia práctica de investigar, tampoco de los lineamientos o reglamentos de titulación. Por el contrario, tiene su origen en el ámbito de la lógica (Jesús Domínguez y Legal, s.f.). La tesis es el texto que se presenta para cumplir con el requisito de la prueba escrita del examen profesional. Independientemente de las especificaciones fijadas por los consejos técnicos o académicos de cada facultad, puede afirmarse que en lo general las tesis de licenciaturas y maestrías cumplen la función de acreditar los conocimientos del sustentante, su capacidad para aplicarlos y su criterio profesional («Como hacer una tesis» 1994).

Un Trabajo de Diploma o tesis es una afirmación cuya veracidad ha sido argumentada, demostrada o justificada de alguna manera. Generalmente enuncia una proposición científica, un axioma o un hecho demostrable. Su objetivo básico es el de informar a través de un soporte material, a un jurado calificador, acerca de un nuevo conocimiento que se ha logrado establecer a partir de una investigación. En su objetivo se encuentra su razón de ser, es necesario cumplir con lo establecido por una disciplina para recibir un título, bien sea este profesional, de licenciatura o doctoral ya que permite poner en práctica los conocimientos adquiridos en clase y obtener mejores resultados (Vega Almeida y Estudio, 2001).

La elaboración de la tesis trae consigo el análisis de teorías, procesos, fenómenos de las ciencias. Los

resultados de dichas investigaciones apoyados en las diferentes técnicas y herramientas, fundamentan la elaboración de nuevas metodologías, planes de acción, propuestas de solución y mejora para el desarrollo de la sociedad. Se considera un ejercicio de investigación con un alto rigor científico - técnico, que le ofrece al estudiante una mayor asimilación de un universo más completo, apropiándose de nuevos conocimientos sobre la base de lo conocido y presentando nuevos aportes que contribuyen al desarrollo de la producción científica. La tesis o Trabajo de Diploma constituye un trabajo académico, resultado de un proceso de formación, que se inicia desde el ingreso del estudiante a la universidad y se liga necesariamente a la investigación.

La importancia de realizar una tesis radica en que ofrece al universitario la oportunidad de efectuar una lectura de la realidad que rompa con los parámetros de una sola visión implicando el desafío de reconocer la necesidad de desarrollar una respuesta ante una situación problemática que lleva consigo el riesgo de equivocarse. El proceso de formación que se lleva a cabo en la universidad debe posibilitar que el estudiante desarrolle formas de pensar, la realidad desde otros ángulos de manera tal que no quede atrapado en una sola lógica de pensamiento. Ello se expresa en actos de conciencia crítico constructivos de nuevas realidades.

En Cuba el Trabajo de Diploma o Tesis de grado es la forma que adopta el ejercicio final de varias carreras y es reconocido por el Reglamento del Trabajo Docente y Metodológico de la Educación Superior. En este Reglamento, se define como un tipo de evaluación de la culminación de los estudios cuyo objetivo es comprobar el grado de dominio de los estudiantes de los objetivos generales de la carrera, mediante la solución, con independencia y creatividad, de un problema propio de la profesión, utilizando la metodología de la investigación científica. El Reglamento especifica los tipos de evaluación donde plantea en su artículo 308.1 que la evaluación del aprendizaje se estructura de forma frecuente, parcial, y de culminación de estudios, en correspondencia con el grado de sistematización de los objetivos a lograr por los estudiantes en cada momento del proceso. Estas formas en su conjunto, caracterizan a la evaluación como un sistema. En el artículo 319.1 plantea que la evaluación de la culminación de los estudios comprueba el cumplimiento de los objetivos generales del plan de estudio (Educación Superior, 2022).

Se hace referencia en el reglamento citado los tipos de evaluación de culminación de estudios, estos son:

- El examen estatal
- La defensa de trabajo de diploma

En el desarrollo de los trabajos de diploma, los profesores deben promover la iniciativa, la independencia y la creatividad de los estudiantes, garantizando el trabajo individual y estimulando el análisis interdisciplinario en la resolución del problema objeto de la investigación. Los profesores que dirigen los trabajos de curso y los trabajos de diploma son designados por el decano de la facultad - carrera, por el jefe del departamento - carrera o por los directores de los centros universitarios municipales o filiales, según el caso, garantizando que tengan la preparación adecuada en correspondencia con los objetivos y la complejidad del tema que desarrollarán los estudiantes. Pueden realizar esta labor también, especialistas reconocidos de las entidades laborales de base, las unidades docentes y centros de investigación (ibíd.).

En la UCI, a los estudiantes de años terminales se le asigna un tema a desarrollar a los cuales se le asigna al menos un tutor. Para el inicio de los trabajos de diplomas se crea un perfil de tesis desarrollado entre el

estudiante y el o los tutores el cual es defendido en un tribunal, dicho tribunal evalúa si el tema propuesto tiene alcance para poder evaluarse como una tesis. Una vez aprobado el tema de tesis el estudiante debe de presentarse a los cortes planificados por la facultad para evaluar el avance del mismo. Una vez realizado los cortes de tesis se planifica la predefensa de la tesis donde se evalúa el cumplimiento de todos los señalamientos en los cortes y se debe de cumplir que la tesis esté terminada completamente, luego de realizar la predefensa se realiza la defensa del trabajo de diploma donde se evalúa al estudiante y se le otorga una calificación.

Todo este proceso se realiza de forma engorrosa, donde por cada corte evaluativo hay que realizar un conjunto de documentos, para evidenciar el nivel de avance de cada estudiante, de la misma forma ocurre en la predefensa. A todo esto, se le puede agregar la asignación de tutores, la conformación de los tribunales que evaluarán los trabajos de diploma, por lo que la gestión de la información se vuelve muy engorrosa.

1.1.2. Gestión de información

La gestión de información es la denominación convencional de un conjunto de procesos por los cuales se controla el ciclo de vida de la información, desde su obtención (por creación o captura), hasta su disposición final (su archivo o eliminación). Tales procesos también comprenden la extracción, combinación, depuración y distribución de la información a los interesados. El objetivo de la gestión de información es garantizar la integridad, disponibilidad y confidencialidad de la información (Barzaga-Sablón; Pincay; Nevárez-Barberán y Cobeña, 2019). Según la revista informática-tecnológica cubana de Joven club la gestión de información es el proceso de controlar, almacenar y recuperar la información adquirida por una entidad, a través de diferentes fuentes. De igual modo pone en uso los recursos de información de la organización, de origen interna como externa para operar, aprender y adaptarse a los cambios del ambiente (JovenClub, 2020). Se concluye que gestión de información es el control, almacenamiento y recuperación de información por parte de una entidad para garantizar la integridad, disponibilidad y confidencialidad de la misma.

Para una correcta gestión de la información son empleados los sistemas informáticos, los cuales son herramientas que son basadas en el diseño de gestión del conocimiento que propone la codificación del conocimiento explícito y la difusión y socialización del conocimiento tácito dentro de las organizaciones (M. D. Pérez, 2017). Se basan en la recopilación y presentación de datos. Su propósito es hacer que la toma de decisiones por parte de los especialistas sea más eficiente y productiva. Por lo tanto, un sistema de gestión de la información puede considerarse como un instrumento que permite la gestión de los recursos de información de una organización. Su finalidad es generar servicios que respondan a las necesidades de los usuarios, aprovechando al máximo sus recursos en función de la mejora continua y de la toma de decisiones.

1.2. Análisis de las soluciones existentes

En esta sección se realizará un análisis de soluciones similares a la que se propone, a fin de obtener las características principales de estos sistemas, los elementos claves que no pueden faltar, las arquitecturas

más utilizadas y en las condiciones en las que se usan, así como cuestiones específicas de cada solución dentro del campo de acción de la presente investigación.

1.2.1. Ámbito internacional

Sistema Informático de Estudios de Postgrado (SIDEP) de la Universidad Central de Venezuela

El sistema permite obtener información sobre la producción intelectual de sus postgrados, accediendo al Catálogo de Trabajos de Grado de Maestría y Tesis Doctorales. El sistema permite realizar consultas por Facultad, curso y año, por autor y por palabras claves. El sistema muestra los datos esenciales de las tesis (autor, título, facultad, curso), así como un resumen de los textos (Venezuela, 2022).



Figura 1.1. Interfaz del sistema SIDEP

Teseo

Sistema de gestión académica encargado de gestionar los resúmenes de las tesis doctorales, posee una base de datos que mantiene el registro de las tesis doctorales declaradas aptas. Todo el resultado del módulo de graduación es un fichero de texto que se envía por correo y que es procesado por el Consejo de Universidades para incorporar los datos a una base de datos temporal, y una ficha impresa que se envía por correo. El Consejo coordina los datos de la ficha con los del fichero, los pasa de la base de datos temporal a la base de datos definitiva, momento en que estarán disponibles para su consulta (Coordinación Universitaria, 2005). El sistema Teseo cuenta con las siguientes funcionalidades:

- Permite añadir, modificar y consultar los resúmenes de las tesis doctorales.
- Genera un fichero con los datos de la universidad a la que pertenece el estudiante y los datos de la dirección de la universidad.
- Imprime la ficha generada con los datos de los estudiantes.



Figura 1.2. Interfaz del sistema Teseo

1.2.2. Ámbito nacional

Sistema de Gestión de la Nueva Universidad (SIGENU)

Sistema informático distribuido del tipo cliente-servidor, construido sobre la base de la utilización de software libre, con el uso de Java como lenguaje de programación. Surge en el año 2004 como una iniciativa del MES. Constituye la principal línea de desarrollo para la informatización de la gestión de la información en la educación superior en Cuba. Actualmente el SIGENU cuenta con una base de datos de más de 120 tablas donde se almacenan todos los datos necesarios para el funcionamiento del sistema (Rajme; Guerra y García, s.f.).

Actualmente posee seis módulos en funcionamiento (CUJAE, 2022):

- Secretaría para el control docente.
- Estadística para la toma de decisiones.
- Profesor para la gestión de evaluaciones.
- Administrador para la gestión de los usuarios.
- Archivo Histórico para la gestión de los egresados y bajas.
- Seguridad para el control y seguimiento.

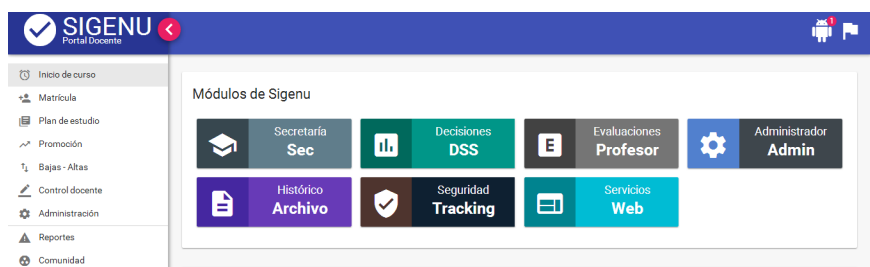


Figura 1.3. Interfaz del sistema SIGENU

Sistema de Gestión Académica (Akademos)

Surge como respuesta a la necesidad de sustentar y dar soporte en la UCI a toda la labor del personal de secretaría. Esta herramienta multiplataforma contribuye al perfeccionamiento de los procesos académicos de cualquier institución. Su uso permite el desarrollo coherente de una estrategia organizacional que articule todos los niveles de decisión presentes en los procesos universitarios. Todos los roles del proceso educativo están involucrados en la solución, por lo que permite el acceso a la información de forma segura a todos los niveles, facilitando la toma de decisiones (UCI, 2022).

Akademos posibilita un mejor control e incidencia en el aprendizaje de los estudiantes, pues con la información que los profesores introducen en tiempo real, es posible determinar el estado de los estudiantes. Además, sus reportes pueden servir para efectuar investigaciones sociales importantes, dada la variedad en términos de proveniencia geográfica, social y académica de la masa estudiantil en la UCI (Costa, 2007).

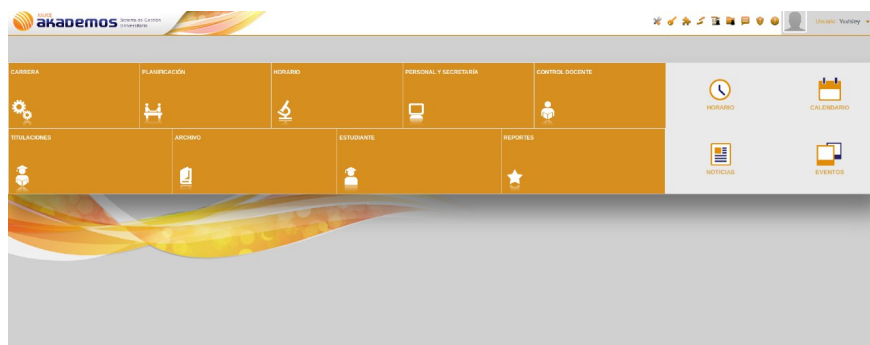


Figura 1.4. Interfaz del sistema Akademos

Repositorio Digital (REPXOS 4.0)

REPXOS 4.0 es un sistema para repositorios digitales desarrollado en la UCI que tiene gran importancia para la preservación y conservación del patrimonio científico de las instituciones; además de que ofrece múltiples beneficios, tanto para la propia institución como para los investigadores. Contempla la incorporación de documentos, proceso de revisión de los mismos, creación de comunidades y colecciones de documentos de acuerdo a las formas de organización de la institución, manejo de cuentas de usuario a través de co-

repositorio electrónico, búsqueda y recuperación de la información, estadísticas. El sistema permite a los usuarios realizar búsquedas avanzadas por campos, palabras clave y a texto completo. El módulo de búsqueda e indexación de REPXOS es un API que permite la indexación de contenidos, la regeneración del índice y realizar búsquedas en el contenido, comunidades y colecciones (Labrada Delgado, 2019).

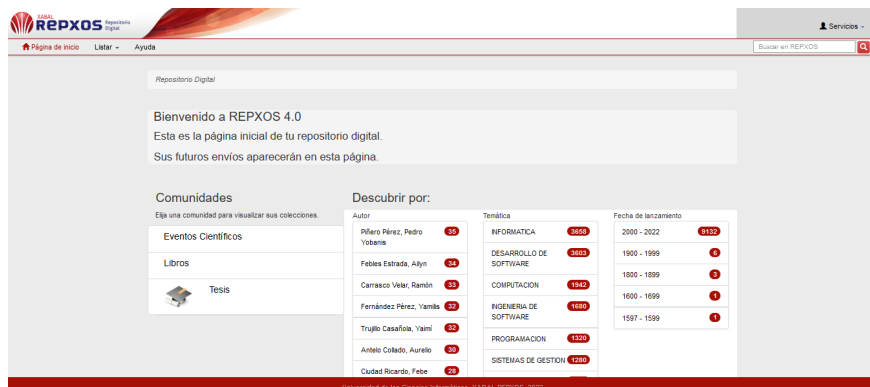


Figura 1.5. Interfaz del sistema REPXOS 4.0

1.2.3. Consideraciones del estudio de los sistemas

A partir de análisis realizado a los sistemas de gestión se evidencia que tanto los sistemas internacionales, como los sistemas nacionales no cumplen las demandas del cliente y por lo tanto no le da solución al problema de la investigación. En el caso de TESEO y el SIDEPE están desarrolladas con tecnologías privativas, lo que va en contra del proceso de migración al software libre que se lleva a cabo en la UCI. Por su parte, SIGENU y Akademos solo se encargan de gestionar el proceso docente de los estudiantes, en el caso de REPXOS 4.0 no contempla del todo la gestión de los trabajos de diplomas, al ser un repositorio no permite funcionalidades deseadas por el cliente. Por lo antes mencionado, es necesario desarrollar un sistema que cumpla con las demandas del cliente y el dominio del problema.

1.3. Análisis de las metodologías, herramientas y tecnologías para el desarrollo del sistema

En esta sección se realizará una descripción de las principales herramientas y tecnologías que se usarán en el desarrollo del sistema para dar solución a la investigación. También se explicará la metodología de desarrollo de software que se tendrá en cuenta, así como la explicación de la razón de su elección.

1.3.1. Metodología

Una metodología de desarrollo de software es un marco de trabajo que se usa para estructurar, planificar y controlar el proceso de desarrollo de sistemas de información. Una gran variedad de estos marcos de trabajo

ha evolucionado durante los años, cada uno con sus propias fortalezas y debilidades.

Una metodología de desarrollo de sistemas no tiene que ser necesariamente adecuada para usarla en todos los proyectos. Cada una de las metodologías disponibles es más adecuada para tipos específicos de proyectos, basados en consideraciones técnicas, organizacionales, de proyecto y de equipo. Por lo es de gran importancia la realización de un análisis para elegir la que resulte mejor para el proyecto en cuestión (Maida y Pacienza, 2015).

Existen dos tipos de metodologías, las tradicionales y las ágiles (Sangama Oñate, 2020). Las tradicionales se componen de numerosa documentación antes de empezar el proceso de implementación, teniendo mucho más peso del análisis y el diseño sobre el proyecto. Además, estas son más convenientes para proyectos grandes o cuyo rendimiento y nivel de calidad son críticos para el éxito de éste. Mientras que las ágiles permiten la comunicación cara a cara entre los miembros del proyecto y los clientes, además ofrecen una buena solución para proyectos de corta duración donde los requisitos no se conocen con exactitud son considerablemente prácticas, ya que no tienen en cuenta gran parte de la documentación y capacitadas para utilizarse en proyectos en los que los requisitos funcionales suelen variar con frecuencia durante el proceso de desarrollo del software (Montero; Cevallos y Cuesta, 2018; Al-Saqqa; Sawalha y AbdelNabi, 2020).

La tabla 1.1 muestra una comparación entre ambas metodologías (tradiciones y ágiles).

Tabla 1.1. Diferencias entre metodologías ágiles y tradicionales (Fuente: (ibíd.)).

Metodologías ágiles	Metodologías tradiciones
Basadas en heurísticas provenientes de prácticas de producción de código	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Especialmente preparados para cambios durante el proyecto	Cierta resistencia a los cambios
Impuestas internamente (por el equipo de desarrollo)	Impuestas externamente
Proceso menos controlado, con pocos principios	Proceso mucho más controlado, con numerosas políticas y normas
No existe contrato tradicional o al menos es bastante flexible	Existe un contrato prefijado
El cliente es parte del equipo de desarrollo	El cliente interactúa con el equipo de desarrollo mediante reuniones
Grupos pequeños (menos de 10 integrantes) y trabajando en el mismo sitio	Grupos grandes y posiblemente distribuidos
Pocos artefactos	Más artefactos
Pocos roles	Más roles
Menos énfasis en la arquitectura del software	La arquitectura del software es esencial y se expresa mediante modelos

Para el desarrollo del sistema informático, se parte de un enfoque ágil pues se trata de un proyecto que posee un equipo de desarrollo pequeño con poca experiencia, con el cliente formando parte de él para convenir los

detalles de los requisitos y así poder implementarlos, probarlos y validarlos.

Programación Extrema (XP, por sus siglas en inglés)

Dentro de las metodologías ágiles, para guiar el desarrollo de la solución propuesta se hace uso de la metodología XP, por ser una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo.

XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. Los principios y prácticas son de sentido común pero llevadas al extremo y de ahí proviene su nombre (Sangama Oñate, 2020).

Es una de las metodologías de desarrollo de software más exitosas en la actualidad, utilizada para proyectos de corto plazo, equipos pequeños y cuyo plazo de entrega está definido. La metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto (Saleh; Huq y Rahman, 2019).

1.3.2. Herramientas y tecnologías

Herramientas de modelado

- **Visual Paradigm:** Es una suite de herramientas Ingeniería de Software Asistida por Computadora (CASE, por sus siglas en inglés) para el desarrollo de aplicaciones usando Lenguaje de Modelado Unificado (UML, por sus siglas en inglés), perfecto para la construcción de sistemas a gran escala. Aporta al proceso de desarrollo calidad, confiabilidad y estabilidad, así como soporte para la Programación Orientada a Objeto (POO). Es perfecta para la construcción de artefactos, necesarios durante todo el proceso ingenieril del desarrollo, pues permite generar muchos de forma automatizada (Fonseca Aristigüi y Sanabria Balber, 2017; Paradigm, 2022).
- **UML:** Es un lenguaje estándar para escribir diseños de software, permite visualizar, especificar, construir y documentar los artefactos de un sistema de software. Al igual que los arquitectos de edificios crean planos, los arquitectos de software usando como herramienta las prestaciones de UML, pueden crear diagramas para ayudar a los desarrolladores de software a su construcción. Además de permitir socializar y especificar el diseño de un sistema (Pressman, 2005).

Lenguajes de programación

- **Python:** Según Van Rossum et al., es un lenguaje de programación diseñado por Guido van Rossum, que se caracteriza por ser de alto nivel, interpretado, de propósito general y sigue una filosofía que enfatiza la limpieza y la fácil comprensión, con el uso de indentación y la ausencia de elementos que

están presentes en otros lenguajes que dificultan su entendimiento. Además de soportar múltiples paradigmas de programación entre los que están el procedural, el funcional y la POO.

Está diseñado para resolver problemas reales del mundo de la programación, lo que se ve potenciado con su flexibilidad y el hecho de ser multiplataforma y de tipado dinámico, permitiendo a los programadores hacer su trabajo con mayor libertad, fomentando de esta manera su creatividad a la hora de encontrar soluciones (Srinath, 2017).

Según un estudio del departamento de ingeniería electrónica y computacional de la facultad de ingeniería y electrónica de la universidad del Estado de Kwara en Nigeria, el lenguaje de programación Python es menos complejo que el lenguaje de programación C++, luego de efectuar una comparativa de los dos, en cuanto a una métrica denominada complejidad de software, resaltando la importancia de que los desarrolladores de software tomen siempre en cuenta el hecho de que elegir un lenguaje de programación poco complejo, facilitará la administración y el mantenimiento del sistema informático que se desee crear (Balogun, 2022).

Resumen de sus principales características según (Srinath, 2017):

- No es fácil de mantener, es necesaria experiencia para escribir código mantenible.
- Lento comparado con otros lenguajes, debido a que es muy flexible.
- Gran cantidad de librerías disponibles para resolver problemas.
- Código abierto.
- Fácil de combinar con otros lenguajes.
- Puede ser usado para realizar una gran variedad de programas.
- Posee una comunidad inmensa.
- Simple y fácil de entender.

Este lenguaje de programación es junto con las tecnologías de JavaScript uno de los más usados para el desarrollo web en el lado del servidor, aunque en el campo del big data y aprendizaje automático se está volviendo cada vez más popular, no solo porque su extenso paquete de librerías para ello lo respalde, sino porque puede ser fácilmente integrado con las aplicaciones web.

- **JavaScript:** Es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario. Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios (J. E. Pérez, 2019).

Este lenguaje fue creado con bajas expectativas, sin embargo, en poco tiempo superó a Java como el lenguaje principal para crear páginas web dinámicas. Aunque los primeros 20 años de JavaScript se caracterizaron por fallidos intentos de mejorarlo, rediseñarlo o reemplazarlo, al final de ese período se convirtió en el lenguaje de programación más difundido en el mundo no solo para páginas web,

adicionándose la posibilidad de construir aplicaciones web usando Node.js, así como aplicaciones de escritorio y para teléfonos móviles, con el uso de React Native, esto sin dejar de hablar de su uso en robots y sistemas embebidos (Wirfs-Brock y Eich, 2020).

- **Leguaje de marcado de hipertexto (HTML, por sus siglas en inglés):** Es un lenguaje de marcado o etiquetas que se emplea para darle formato a los documentos o páginas web que se publican en Red Informática Mundial (WWW, por sus siglas en inglés). Fue desarrollado por Tim Berners-Lee basándose en el estándar El Lenguaje de marcado generalizado estándar (SGML, por sus siglas en inglés). La organización El Consorcio de la Red Informática Mundial (W3C, por sus siglas en inglés) se encarga de su estandarización, para así reducir los problemas de compatibilidad con los navegadores web, que son las aplicaciones que se usan para su interpretación y comprensión humana. En su versión más reciente Leguaje de marcado de hipertexto Versión 5 (HTML5, por sus siglas en inglés), se le da mucha importancia a la semántica de las etiquetas, ayudando con esto a una mejor interpretación de los navegadores y los motores de búsqueda en donde se encuentren indexadas (Luján Mora, 2016), (Consortium et al., 1999) y (Malqui, 2018).
- **Hoja de estilo en Cascada (CSS, por sus siglas en inglés):** El CSS es un lenguaje de estilos estandarizado por W3C y utilizado para definir la presentación, el formato y la apariencia de un documento de marcado como por ejemplo HTML. Nacieron con la necesidad de diseñar la presentación de la información de tal manera que resulte más comprensible, además de permitir facilitar dicho proceso y poderlo hacer para ofrecer distintas presentaciones de la información en dispositivos diferentes con características particulares a cada uno. En su última versión Hoja de estilo en Cascada Versión 3 (CSS3, por sus siglas en inglés), se ofrecen nuevas facilidades y herramientas que permiten al diseñador o programador presentar los contenidos de forma más vistosa, en menos tiempo y con menos esfuerzo, por lo que resulta una herramienta muy atractiva (Puig, 2013) y (Malqui, 2018).

Entorno de Desarrollo Integrado (IDE, por sus siglas en inglés)

- **Microsoft Visual Studio Code:** Es un editor de código ligero y adaptable desarrollado por Microsoft, puede correr sobre Windows, macOS y Linux. Tiene soporte para JavaScript, TypeScript and Node.js, pero usando su rico ecosistema de extensiones se puede personalizar a niveles importantes, permitiendo construir un entorno de desarrollo ideal para cualquier programador, en casi cualquier lenguaje de programación, agregándole capacidades que rivalizan con cualquier IDE especializado (Microsoft, 2022).

Framework

- **Django:** Según D. Rubio es un entorno de programación en Python diseñado para el desarrollo de aplicaciones web del lado del servidor. Es un framework sencillo de instalar, posee una documentación muy completa y una amplia comunidad. Esta herramienta está basada en la arquitectura Modelo Vista Controlador (MVC, por sus siglas en inglés), lo que se denomina Modelo Vista Plantilla (MTV, por

sus siglas en inglés), entre sus filosofías está la de la creación de aplicaciones complejas y robustas en poco tiempo, garantizando siempre la seguridad, y basada siempre en No te repitas (DRY, por sus siglas en inglés). Está diseñado para trabajar con bases de datos relacionales, aunque su flexibilidad le permite con las no relacionales también, además de ser utilizado para aplicaciones de comunicación en tiempo real por internet, así como integrada con las prácticas modernas de JavaScript.

El éxito de Django queda evidenciado en la gran cantidad de ejemplos de organizaciones importantes del mundo que adoptan esta tecnología, tales como Pinterest, Instagram, Public Broadcasting System (PBS, por sus siglas en inglés) y National Geographics, y principalmente se debe a su capacidad de optimizar el tiempo de desarrollo de sistemas complejos, pues con sus principios de diseño produce uno de los procesos productivos más eficientes comparados con otros frameworks para el desarrollo web.

Cuenta con un lenguaje de plantillas, que permite separar la lógica de presentación de la lógica del negocio posibilitando numerosas ventajas para el proyecto:

- Los diseñadores no pueden afectar el código de programación y tienen total libertad a la hora del diseño sin necesidad de molestar al programador.
- Los programadores pueden efectuar cambios en el código de la aplicación sin afectar la parte visual.
- Se genera un código más limpio y legible.

Posee en su núcleo un Mapeador de Objeto Relacional (ORM, por sus siglas en inglés) que permite crear bases de datos y realizar consultas sin usar Lenguaje de Consultas Estructurado (SQL, por sus siglas en inglés), sino aplicando código Python usando las librerías de Django para ello. Esto posibilita que en caso de que sea necesario cambiar de tipo de base de datos o de gestor se hace extremadamente fácil, pues solo hay que cambiar algunas opciones en el archivo de configuración del framework.

Django posibilita definir proyectos y aplicaciones para sistemas de información web. Cuando se define un proyecto Django, automáticamente se genera una serie de archivos, y uno de los más relevantes es (`settings.py`). El archivo `setting.py` se utiliza para definir variables de configuración del sistema web tales como las aplicaciones que incluye, parámetros de conexión a base de datos, idioma, entre otros. Una vez que se define un proyecto Django, es posible definir aplicaciones para dicho proyecto. Para cada aplicación de proyecto Django, se pueden definir modelos que representan tablas en la base de datos establecida (archivo `models.py`) junto con parámetros de visualización de estos modelos, los cuales permiten mediante el ya mencionado ORM generar las tablas de la base de datos y las consultas. Además, en un proyecto Django la capa plantilla se define con archivos con extensión `.html` para la visualización de datos (archivos `.html` en carpeta `templates`). Los modelos que se usarán en una aplicación deben estar incluidos en el (archivo `admin.py`) (Vidal-Silva; Sánchez-Ortiz; Serrano y J. M. Rubio, 2021).

Al ser herramientas de desarrollo del mundo libre o código abierto, además de su simplicidad para desarrollar soluciones, hace que muchos desarrolladores en el mundo usen y desarrollen extensiones

para estas herramientas.

Sistema Gestor de Base de Datos (SGBD)

- **PostgreSQL:** Los SGBD son sistemas que generan independencia de datos, gestionan la concurrencia de acceso, garantiza la integridad de los datos, proveen la persistencia de los mismo y brindan la seguridad necesaria a toda la información que se encuentra almacenada en las bases de datos Pilicita Garrido; Borja López y Gutiérrez Constante, 2021.

PostgreSQL es un SGBD de tipo relacional y código libre, que posee una comunidad activa y en constante desarrollo, que da la posibilidad de crear tipos de datos definidos por el programador y posee una alta compatibilidad con el framework Django Gómez Bermejo, 2018. Se tiene buen criterio de este gestor en cuanto a su estabilidad, potencia, robustez y la facilidad de administración e implementación.

Bibliotecas

- **JQuery:** Es una biblioteca de JavaScript software libre y de código abierto, que permite simplificar el trabajo con JavaScript. Es rápida, sencilla, eficaz y poderosa, se usa para darle dinamismo y funcionalidades al frontend de las páginas web, mediante la manipulación del DOM y el manejo de eventos o acciones del usuario, facilita la aplicación de animaciones y la utilización de AJAX para simular el comportamiento de una aplicación de escritorio. El hecho de simplificar las funcionalidades del lenguaje garantiza lograr mejores resultados con menor tiempo y esfuerzo (Ventura Bautista, 2021).
- **Bootstrap:** Bootstrap es una biblioteca, dirigido al front-end de una aplicación web, por lo que está orientado a la parte que se ejecuta en el cliente, en este caso el navegador web. Utiliza HTML5, CSS y JavaScript, los lenguajes por excelencia para contenido web, para hacer uso de sus librerías que permiten diseñar la web de una manera profesional creando al mismo tiempo un diseño adaptado para dispositivos móviles, con el uso de la filosofía "mobile first" del inglés móvil primero, la cual consiste en enfocarse primero en la vista de teléfonos móviles y luego ir escalando hasta llegar a las vistas de mayor resolución de pantalla. Es código abierto y uno de los proyectos más populares de todos los tiempos en GitHub (Gómez Bermejo, 2018; Niska, 2014) y (Jacob Thornton, 2022).

1.4. Conclusiones parciales

En este capítulo se abordaron los elementos teóricos que sustentan la propuesta solución del problema en cuestión. Esto permite arribar a las siguientes conclusiones:

- La gestión de la información se lleva a cabo manualmente, por lo cual se definió como propuesta solución un sistema informático que contribuya a la gestión de la información asociada a los trabajos de diploma en la Facultad 4 de la Universidad de las Ciencias Informáticas.

- El análisis de sistemas homólogos permitió identificar características necesarias en la aplicación y comprender la necesidad de desarrollar un sitio propio que se adapte a las necesidades de la facultad.
- A partir del análisis de las herramientas, tecnologías y metodología se emplea como metodología de desarrollo XP, Django como framework de desarrollo y como lenguajes de programación Python.

Características y diseño del sistema para la gestión de los trabajos de diploma en la Facultad 4 de la UCI

El presente capítulo aborda los principales aspectos relacionados con las características de la propuesta de solución. Se identifican los requisitos funcionales y no funcionales con los que debe cumplir la solución propuesta, así como estilo arquitectónico y los patrones de diseño para lograr buenas prácticas en el diseño y posterior implementación del sistema. Igualmente se muestran los principales artefactos de ingeniería de software propuestos por la metodología utilizada.

2.1. Descripción de la propuesta de solución

A partir del problema y el objetivo planteado en la presente investigación, se propone como solución un sistema informático para la gestión de la información asociada a los trabajos de diplomas. El software se desarrollará sobre la base de una aplicación web posibilitándoles a los usuarios disfrutar de los beneficios que estas aplicaciones reportan, el acceso múltiple al sistema por medio de un servidor web, así como las actualizaciones y los mantenimientos del software sin necesidad de distribuir e instalar. La gestión de la información se realizará a través de una base de datos localizada en el servidor lo que permitirá, realizarla de forma simultánea por varios usuarios.

El sistema contribuye a la gestión de la información que se genera a partir de los ejercicios de culminación de estudios en la UCI. El sistema permite la centralización de la información asociada a los trabajos de diploma permitiendo el almacenamiento de manera organizada, facilitando su búsqueda y visualización. Además de brindar información histórica de los mismos en años anteriores, el sistema cuenta con un conjunto de módulos dedicados a, la gestión de los usuarios la cual permitirá la visualización de las funcionalidades del sistema en correspondencia con los permisos de cada usuario según su rol, la gestión de usuario permitirá otorgarle al sistema un mejor flujo de información; la gestión de trabajo de diploma; la gestión de tribunal; la gestión de banco de problema y la gestión de los reportes asociados a la información almacenada en la base de datos.

2.2. Especificación de requisitos del sistema

Con el objetivo de que el sistema propuesto se desarrolle con las necesidades dictadas por el cliente final a partir del uso de la tormenta de ideas y la aplicación de encuesta, se obtuvieron los requisitos funcionales y no funcionales.

2.2.1. Requisitos funcionales

Teniendo en cuenta las características de la propuesta en cuanto a prestaciones y operatividad se identifican 50 requisitos funcionales. A continuación, la tabla 2.1 muestra los requisitos del sistema.

Tabla 2.1. Descripción de los Requisitos Funcionales. (Fuente: Elaboración propia).

No.	Nombre	Prioridad	Complejidad
RF1	Autenticar usuario	Alta	Media
RF2	Registrar usuario	Alta	Media
RF3	Modificar usuario	Alta	Media
RF4	Eliminar usuario	Alta	Media
RF5	Crear rol	Alta	Media
RF6	Eliminar rol	Alta	Media
RF7	Modificar rol	Alta	Media
RF8	Listar usuario	Alta	Media
RF9	Crear perfil de tesis	Alta	Baja
RF10	Modificar perfil de tesis	Alta	Media
RF11	Aprobar perfil de tesis	Alta	Baja
RF12	Mostrar detalles perfil de tesis	Alta	Media
RF13	Eliminar perfil de tesis	Alta	Media
RF14	Crear problema investigativo	Alta	Baja
RF15	Eliminar problema investigativo	Alta	Media
RF16	Modificar problema investigativo	Alta	Baja
RF17	Mostrar detalles problema investigativo	Alta	Baja
RF18	Convertir problema en perfil	Alta	Media
RF19	Evaluar corte de tesis	Alta	Baja
RF20	Añadir las actas de los cortes de tesis	Alta	Media
RF21	Descargar las actas de los cortes de tesis	Alta	Baja
RF22	Crear acta de predefensa	Alta	Baja
RF23	Eliminar acta de predefensa	Alta	Media
RF24	Modificar acta de predefensa	Alta	Baja
RF25	Descargar de acta de predefensa	Alta	Baja
RF26	Mostrar detalles de acta de predefensa	Alta	Baja
RF27	Crear acta de defensa	Alta	Baja
Continúa en la siguiente página			

No.	Nombre	Prioridad	Complejidad
RF28	Eliminar acta de defensa	Alta	Media
RF29	Modificar acta de defensa	Alta	Baja
RF30	Descargar de acta de defensa	Alta	Baja
RF31	Mostrar detalles de acta de defensa	Alta	Baja
RF32	Buscar perfil de tesis	Alta	Baja
RF33	Buscar problema investigativo	Alta	Baja
RF34	Buscar documento de tesis	Alta	Baja
RF35	Buscar usuario	Alta	Baja
RF36	Generar reporte de la evaluación del corte	Alta	Media
RF37	Generar reporte de perfiles existentes	Alta	Media
RF38	Generar reporte de perfiles aprobados	Alta	Media
RF39	Generar reporte de perfiles sin aprobar	Alta	Media
RF40	Generar reporte de perfiles en revisión	Alta	Media
RF41	Generar reporte de tutores por Departamento	Alta	Media
RF42	Generar reporte de estudiantes por Departamento	Alta	Media
RF43	Adjuntar documento de tesis	Alta	Media
RF44	Descargar documento de tesis	Alta	Media
RF45	Mostrar documento de tesis	Alta	Media
RF46	Crear tribunal de tesis	Alta	Media
RF47	Eliminar tribunal de tesis	Alta	Media
RF48	Modificar tribunal de tesis	Alta	Media
RF49	Mostrar tribunal de tesis	Alta	Media
RF50	Asignar estudiante a tribunal de tesis	Alta	Media

2.2.2. Requisitos no funcionales

- Usabilidad.
 - RnF1. Tipo de aplicación informática: la herramienta debe ser web, que contemple estándares y patrones de diseño de Interfaz de Usuario (UI, por sus siglas en inglés) modernas.
 - RnF2. El sistema debe poseer una UI fácil de utilizar para cualquier tipo de usuario con conocimientos básicos de computación en el manejo de ordenadores.
- Confiabilidad.
 - RnF3. El sistema debe ser tolerante a fallos, y mostrar solo la información necesaria para orientar al usuario.
- Hardware y software requerido para utilizar la aplicación.
 - RnF4. Sistema Operativo: Podrá utilizar cualquier sistema operativo.
 - RnF5. El sistema debe de ejecutarse sobre el navegador Mozilla Firefox o Google Chrome con versiones superiores a 79 y 60 respectivamente.

- RnF6. La computadora que se utilice el cliente debe contar como mínimo con 2 GB de RAM y la que se utilice para el servidor debe contar como mínimo 500 GB de almacenamiento y como mínimo 4 GB de RAM.
- Eficiencia.
 - RnF7. El sistema debe permitir que los usuarios (100) interactúen con él de manera concurrente.
 - RnF8. El tiempo de demora de una petición al servidor debe ser menor a dos (2) segundos
- Seguridad.
 - RnF9. El acceso a la información debe estar restringido por usuario, contraseña y rol.
 - RnF11. Cuando un usuario se autentique o registre en el sistema se le brindará la información correspondiente con su rol.
- Restricciones de diseño e implementación.
 - RnF12. El marco de trabajo que se utilizará está basado principalmente en el uso de herramientas no privativas:
 - Lenguaje de programación del lado del servidor Python v3.10.5.
 - Lenguaje de programación del lado del cliente JavaScript ECMAScript2015.
 - Lenguaje para el estilo del documento web CSS3.
 - Lenguaje para la estructuración del documento web HTML5.
 - Lenguaje para el modelado del proceso UML v2.5.1.
 - Framework de desarrollo web Django v4.0.1
 - Framework para los estilos Bootstrap v5.1.
 - Herramienta para el modelado del proceso Visual Paradigm v15.1 .
 - Herramienta para la codificación Microsoft Visual Studio Code v1.68.
 - Herramienta para la realización de pruebas de seguridad Acunetix Web Vulnerability Scanner v9.5.
 - RnF13. Se utilizará como metodología de software XP.

2.3. Descripción de las historias de usuario

Para el encapsulamiento de los requisitos del software, tal como lo plantea la metodología XP, se generaron un total de 50 HU, a continuación, se muestran las correspondientes a los requisitos funcionales **Crear perfil de tesis, Modificar perfil de tesis, Eliminar perfil de tesis, Crear tribunal de tesis y Asignar estudiante a tribunal de tesis**, las restantes se pueden encontrar en el anexo .3.

Tabla 2.2. Historia de usuario # 1

Historia de usuario

Continúa en la próxima página

Tabla 2.2. Continuación de la página anterior

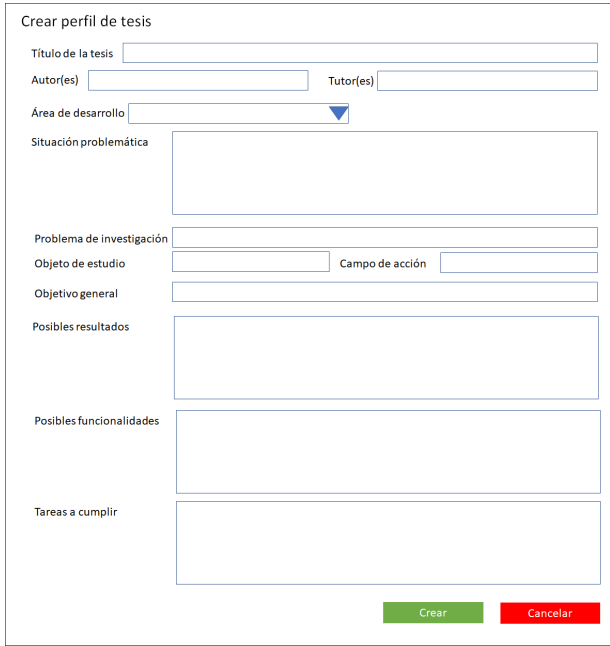
Número: 1	Nombre: Crear perfil de tesis
Usuario: Usuario	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 9
Programador responsable: Abel Machado Montes	
Descripción: El sistema debe permitir que el usuario pueda crear un perfil de tesis mostrándole una interfaz donde pueda llenar los campos solicitados (título, autor(es), tutor(es), clasificación del área de desarrollo, situación problemática, problema de la investigación, objeto de estudio, objetivo general, campo de acción, posibles resultados, posibles funcionalidades y tareas a cumplir). El sistema debe validar que se encuentre todos los campos llenos y correctos.	
Observaciones: Usuario autenticado y con los permisos	
Interfaz: 	

Tabla 2.3. Historia de usuario # 2

Historia de usuario	
Número: 2	Nombre: Modificar perfil de tesis
Usuario: Usuario	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 10
Programador responsable: Abel Machado Montes	

Continúa en la próxima página

Tabla 2.3. Continuación de la página anterior

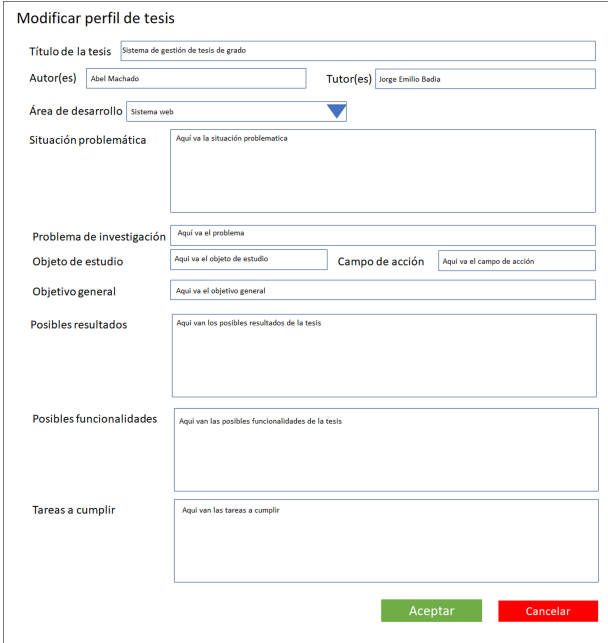
Descripción: El sistema debe permitir que el usuario pueda modificar un perfil de tesis mostrándole una interfaz donde pueda modificar los campos (título, autor(es), tutor(es), clasificación del área de desarrollo, situación problemática, problema de la investigación, objeto de estudio, objetivo general, campo de acción, posibles resultados, posibles funcionalidades y tareas a cumplir). El sistema debe validar que se encuentre todos los campos llenos y correctos.	
Observaciones: Usuario autenticado y con los permisos	
Interfaz: 	

Tabla 2.4. Historia de usuario # 3

Historia de usuario	
Número: 3	Nombre: Eliminar perfil de tesis
Usuario: Usuario	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 13
Programador responsable: Abel Machado Montes	
Descripción: El sistema debe permitir que el usuario pueda eliminar un perfil de tesis ya registrado, mostrándole un mensaje de si está seguro que desea eliminar. El sistema muestra en pantalla dos opciones una de “Aceptar” y otra de “Cancelar”. Si el usuario presiona la opción de Aceptar el perfil automáticamente es eliminado de la base de datos y en el caso de Cancelar, no se elimina el perfil y regresa a la interfaz de listar perfiles de tesis	
Observaciones: Usuario autenticado y con los permisos	

Continúa en la próxima página

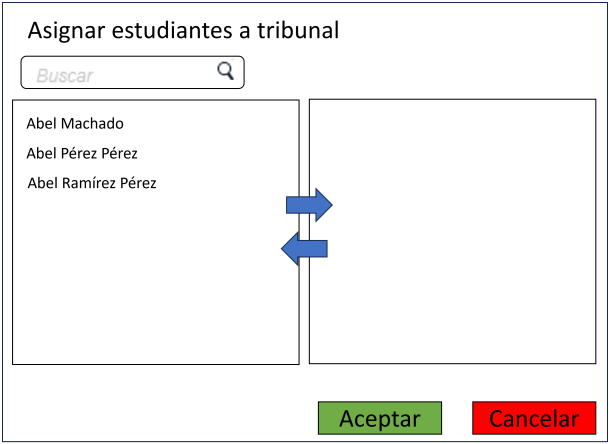
Tabla 2.4. Continuación de la página anterior

Interfaz:	¿Está seguro que desea eliminar el perfil de tesis seleccionado? AceptarCancelar
------------------	---

Tabla 2.5. Historia de usuario # 4

Historia de usuario	
Número: 4	Nombre: Crear tribunal de tesis
Usuario: Administrador, profesor	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 46
Programador responsable: Abel Machado Monteso	
Descripción: El sistema debe permitir un crear un tribunal de tesis. Para ello el usuario debe seleccionar de los profesores existentes en base de datos quien tiene el rol de presidente, el de secretario y el de vocal.	
Observaciones: Usuario autenticado y con los permisos necesarios, y los profesores deben estar registrados en el sistema	
Interfaz: Crear de Tribunal de tesis Identificador: Presidente: Elige ▼ Secretario: Elige ▼ Vocal: Elige ▼ AceptarCancelar	

Tabla 2.6. Historia de usuario # 5

Historia de usuario	
Número: 5	Nombre: Asignar estudiante a tribunal de tesis
Usuario: administrador, profesor	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 50
Programador responsable: Abel Machado Monteso	
Descripción: El sistema debe permitir asignarle estudiantes a un tribunal de tesis ya registrado en la base de datos. El usuario tiene la opción de seleccionar al estudiante que desea asignar a un tribunal y presiona la opción de “Asignar”, el estudiante automáticamente pasa a formar parte del tribunal asignado.	
Observaciones:	
Interfaz: 	

2.4. Planificación de la entrega

Después de tener ya definidas las HU es necesario crear un plan de publicaciones donde los desarrolladores y clientes establecen los tiempos de implementación ideales de las HU, la prioridad con la que serán implementadas y las historias que serán implementadas en cada versión del programa (Centellas Coarite, 2015).

2.4.1. Estimación del esfuerzo por Historias de Usuario

A continuación, se muestra la tabla 2.7 que encierra los resultados obtenidos de la estimación del esfuerzo para cada una de las HU identificadas en el proyecto en desarrollo.

CAPÍTULO 2. CARACTERÍSTICAS Y DISEÑO DEL SISTEMA PARA LA GESTIÓN DE LOS TRABAJOS DE DIPLOMA EN LA FACULTAD 4 DE LA UCI

Número de HU	HU	Puntos Estimados
--------------	----	------------------

Tabla 2.7. Estimación de esfuerzo por HU. (Fuente: Elaboración propia).

No. de HU	HU	Puntos Estimados
1	Autenticar usuario	1
2	Registrar usuario	
3	Modificar usuario	
4	Eliminar usuario	
5	Crear rol	
6	Eliminar rol	
7	Modificar rol	
8	Listar usuario	
9	Crear perfil de tesis	1
10	Modificar perfil de tesis	
11	Aprobar perfil de tesis	
12	Mostrar detalles perfil de tesis	
13	Eliminar perfil de tesis	
14	Crear problema investigativo	1
15	Eliminar problema investigativo	
16	Modificar problema investigativo	
17	Mostrar detalles problema investigativo	
18	Convertir problema en perfil	0.5
19	Evaluar corte de tesis	
20	Añadir las actas de los cortes de tesis	
21	Descargar las actas de los cortes de tesis	1
22	Crear acta de predefensa	
23	Eliminar acta de predefensa	
24	Modificar acta de predefensa	
25	Descargar acta de predefensa	
26	Mostrar detalles de acta de predefensa	1
27	Crear acta de defensa	
28	Eliminar acta de defensa	
29	Modificar acta de defensa	
30	Descargar acta de defensa	
31	Mostrar detalles de acta de defensa	0.5
32	Buscar perfil de tesis	
33	Buscar problema investigativo	
34	Buscar documento de tesis	
35	Buscar usuario	
36	Generar reporte de la evaluación del corte	
37	Generar reporte de perfiles existentes	

Número de HU	HU	Puntos Estimados
38	Generar reporte de perfiles aprobados	
39	Generar reporte de perfiles sin aprobar	
40	Generar reporte de perfiles en revisión	
41	Generar reporte de tutores por Departamento	
42	Generar reporte de estudiantes por Departamento	
43	Adjuntar documento de tesis	1.5
44	Descargar documento de tesis	
45	Mostrar documento de tesis	
46	Crear tribunal de tesis	
47	Eliminar tribunal de tesis	
48	Modificar tribunal de tesis	
49	Mostrar tribunal de tesis	
50	Asignar estudiante a tribunal de tesis	

2.5. Plan de iteraciones

Una vez definidas las HU y realizada la estimación del esfuerzo para cada una de ellas se continúa a la etapa de implementación del sistema. En el que se define un plan de iteraciones, donde se especifica la estimación aproximada con que se implementarán las HU organizadas en cada una de las iteraciones. Teniendo en cuenta el esfuerzo asociado y las prioridades del cliente, se precisa una versión que sea de valor para este.

2.5.1. Desarrollo del plan de iteraciones

El plan tiene como objetivo mostrar la duración de cada iteración, así como el orden en que serán implementadas las HU. Se definieron tres (3) iteraciones, una primera iteración que consta de un total de 13 HU, encargada de la gestión de los usuarios y los perfiles de tesis, una segunda con 18 HU encargada de la gestión de las actas, los problemas investigativos y los cortes y una tercera iteración con 19 HU donde se implementaran aquellas HU que tiene como objetivo el trabajo con los datos registrados en la Base de Datos y adjuntar y descargar los archivos.

Tabla 2.8. Pan de iteraciones (Fuente: Elaboración propia).

Iteración	HU	Duración
	Autenticar usuario	0.5
	Registrar usuario	0.2
	Modificar usuario	0.5
	Eliminar usuario	0.2
	Listar usuario	0.2
	Crear rol	0.5

CAPÍTULO 2. CARACTERÍSTICAS Y DISEÑO DEL SISTEMA PARA LA GESTIÓN DE LOS TRABAJOS DE DIPLOMA EN LA FACULTAD 4 DE LA UCI

Iteración	HU	Duración
	Eliminar rol	0.5
	Modificar rol	0.5
	Crear perfil de tesis	0.5
	Modificar perfil de tesis	0.5
	Aprobar perfil de tesis	0.5
	Mostrar detalles perfil de tesis	0.5
	Eliminar perfil de tesis	0.5
2	Crear problema investigativo	0.5
	Eliminar problema investigativo	0.5
	Modificar problema investigativo	0.5
	Mostrar detalles problema investigativo	0.3
	Convertir problema en perfil	0.2
	Evaluar corte de tesis	0.5
	Añadir las actas de los cortes de tesis	0.5
	Descargar las actas de los cortes de tesis	0.2
	Crear acta de predefensa	0.2
	Eliminar acta de predefensa	0.5
	Modificar acta de predefensa	0.5
	Descargar de acta de predefensa	0.2
	Mostrar detalles de acta de predefensa	0.2
	Crear acta de defensa	0.2
	Eliminar acta de defensa	0.2
	Modificar acta de defensa	0.2
	Descargar de acta de defensa	0.2
	Mostrar detalles de acta de defensa	0.2
3	Buscar perfil de tesis	0.5
	Buscar problema investigativo	0.5
	Buscar documento de tesis	0.2
	Buscar usuario	0.2
	Generar reporte de la evaluación del corte	0.2
	Generar reporte de perfiles existentes	0.5
	Generar reporte de perfiles aprobados	0.5
	Generar reporte de perfiles sin aprobar	0.5
	Generar reporte de perfiles en revisión	0.2
	Generar reporte de tutores por Departamento	0.5
	Generar reporte de estudiantes por Departamento	0.2
	Adjuntar documento de tesis	0.2
	Descargar documento de tesis	0.2
	Mostrar documento de tesis	0.2
	Crear tribunal de tesis	0.2
	Eliminar tribunal de tesis	0.2

Iteración	HU	Duración
	Modificar tribunal de tesis	0.5
	Mostrar tribunal de tesis	0.2
	Asignar estudiante a tribunal de tesis	0.5

2.6. Tarjetas CRC

El uso de las tarjetas CRC permiten al programador centrarse y apreciar el desarrollo orientado a objetos olvidándose de los malos hábitos de la programación clásica. Estas tarjetas representan objetos; la clase a la que pertenece el objeto, las responsabilidades u objetivos que debe cumplir el objeto y las clases que colaboran con cada responsabilidad.

La técnica consiste en dibujar una tarjeta por cada clase u objeto, y dividirla en tres zonas: En la parte superior, el nombre de la clase. Debajo, en la parte izquierda, las responsabilidades de dicha clase. Son sus objetivos, a alto nivel. A la derecha de las responsabilidades, los colaboradores, que son otras clases que ayudan a conseguir cumplir a esta con sus responsabilidades (Lara y Figueroa, 2020).

La utilidad de estas tarjetas radica en que facilitan al equipo entender el proceso de diseño en una manera fácil y concisa.

Tabla 2.9. Tarjeta CRC # 1

Tarjeta CRC	
Clase: ViewProfile	
Responsabilidad	Colaboración
• Insertar_perfil • Listar_perfil • Aprobar_perfil • Modificar_perfil • Eliminar_perfil	ViewProfile ViewProblem ViewUser

Tabla 2.10. Tarjeta CRC # 2

Tarjeta CRC	
Clase: ViewTribunal	
Responsabilidad	Colaboración

Continúa en la próxima página

Tabla 2.10. Continuación de la página anterior

• CrearTribunal() • EditarTribunal() • ListarTribunal() • EliminarTribunal() • DetalleTribunal() • GenerarDoc()	ViewTribunal ViewTribunal_Type ViewUser
--	---

Tabla 2.11. Tarjeta CRC # 3

Tarjeta CRC	
Clase: ViewProfile_Tutor	
Responsabilidad	Colaboración
• CrearProfile_Tutor() • EditarProfile_Tutor() • ListarProfile_Tutor() • EliminarProfile_Tutor()	ViewProfile_Tutor ViewProfile

2.7. Estilo arquitectónico

El diseño arquitectónico se interesa por entender cómo debe organizarse un sistema y cómo tiene que diseñarse la estructura global de ese sistema. Es la primera etapa en el proceso de diseño del software. Es el enlace crucial entre el diseño y la ingeniería de requerimientos, ya que identifica los principales componentes estructurales en un sistema y la relación entre ellos. La salida del proceso de diseño arquitectónico consiste en un modelo arquitectónico que describe la forma en que se organiza el sistema como un conjunto de componentes en comunicación (Sommerville, 2011).

Para el diseño de la propuesta de solución se utilizará el estilo Llamada y Retorno. Este estilo permite que los datos sean pasados como parámetros y el manejador principal proporciona un ciclo de control sobre las subrutinas.

2.7.1. Patrón arquitectónico

Un patrón arquitectónico se puede considerar como una descripción abstracta estilizada de buena práctica, que se ensayó y puso a prueba en diferentes sistemas y entornos. De este modo, un patrón arquitectónico debe describir una organización de sistema que ha tenido éxito en sistemas previos. Debe incluir información sobre cuándo es y cuándo no es adecuado usar dicho patrón, así como sobre las fortalezas y debilidades del patrón (ibíd.).

Como propuesta de solución para el desarrollo de la aplicación se determinó el patrón MTV que propone Django como una versión del MVC.

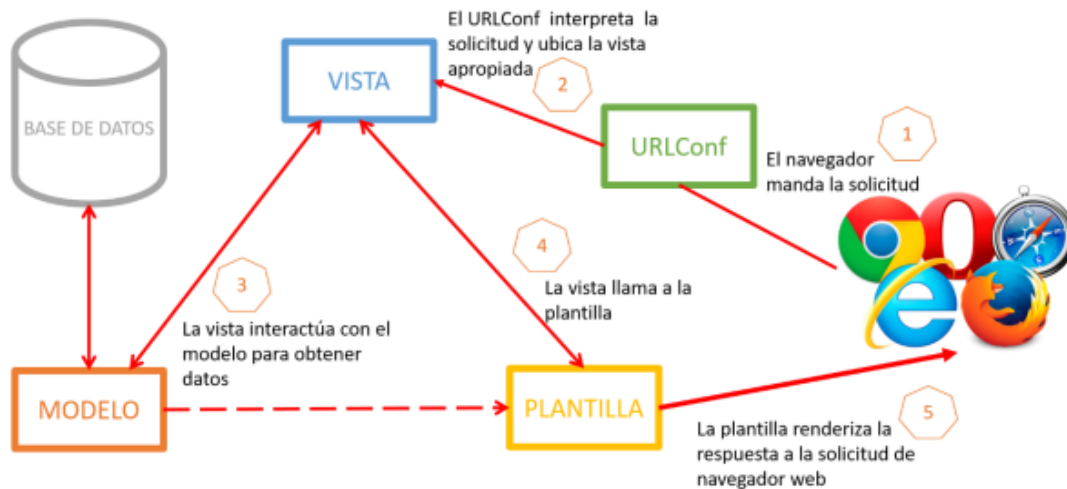


Figura 2.1. Representación de la arquitectura. (Fuente: Elaboración propia))

- **Modelo:** Es la parte del MTV que hace de medio de almacenamiento de datos, donde se puede indicar y controlar el comportamiento de los datos. Representa la información con la que trabaja la aplicación. Este no tiene conocimiento específico de las vistas o de las plantillas, ni siquiera contiene referencias a ellos.
- **Vistas:** Contiene la lógica de la aplicación, la que responde a las peticiones de los usuarios e interactúa con la template y hace uso del modelo de datos para poder acceder a la información almacenada. Es el equivalente del controlador en MVC. Es necesaria para devolver una respuesta hacia el cliente que la solicita, también procesa las peticiones o solicitudes que accederán al modelo para poder entregar u obtener los datos.
- **Plantillas:** Es la interfaz de usuario, es la parte del Framework que permite mostrar los datos almacenados y procesados. Da la posibilidad al usuario de interactuar con el sistema. Decide la forma en la que se presentarán los datos devueltos por la vista en el navegador web utilizando estilos CSS o brindando dinamismo a través de JavaScript.

2.8. Patrones de diseño

Un patrón de diseño es una solución a un problema de diseño que puede ser utilizada de forma repetida, cuya efectividad ya está probada. Puede ser usado en diferentes contextos, no es código de programación sino una descripción de cómo resolver un problema dado. Con el uso de ellos en la confección de una aplicación, se

garantizarán diseños flexibles, reusables y modulares. Cada patrón resuelve un problema en específico, por lo que es importante conocerlos y aplicarlos en la situación en la que lo requiera. Son soluciones bien detalladas que promueven buenas prácticas, facilitando su comprensión y reusabilidad (Gamma; Helm; R. Johnson; R. E. Johnson; Vlissides et al., 1995; Pressman, 2005).

1. Patrones Generales de Software de Asignación de Responsabilidad (GRASP, por sus siglas en inglés): Los patrones GRASP, más que patrones propiamente dichos, son una serie de buenas prácticas de aplicación recomendable en el diseño de software. Entre ellos, los patrones Experto, Creador, Bajo acoplamiento y Alta cohesión guardan directa relación con la creación y asignación de responsabilidades a los objetos (Tabares, 2010).

- Experto: Posibilita una adecuada asignación de responsabilidades facilitando la comprensión del sistema, su mantenimiento y adaptación a los cambios con reutilización de componentes. Trae como beneficios que se mantiene el encapsulamiento de la información puesto que los objetos usan su propia información para llevar a cabo tareas, lo que permite crear sistemas más robustos y mantenibles, se distribuye el comportamiento de las clases que contienen la información requerida, por lo que las clases se vuelven más fáciles de entender y mantener (Larman, 2003). Este patrón se encuentra aplicado en todas las clases del archivo Models.py.

- Creador: Aporta un principio general para la creación de objetos, una de las actividades más frecuentes en programación. Asigna a una clase la responsabilidad de crear instancias de la otra cuando se cumplen determinadas condiciones en su relación, como son que la primera agrega a la segunda, o contiene los datos de inicialización de la segunda o utiliza estrechamente a esa clase para su funcionamiento(ibíd.).

El patrón se hace evidente en las clases contenidas dentro del archivo View.py, las cuales tienen la responsabilidad de instanciar los objetos creados con cada módulo de configuración básica.

- Alta cohesión: Es una medida que determina cuán relacionadas y adecuadas están las responsabilidades de una clase, de manera que no realice un trabajo colosal; una clase con baja cohesión realiza un trabajo excesivo, haciéndola difícil de comprender, reutilizar y conservar. Este patrón expresa que se debe asignar a las clases responsabilidades que trabajen sobre una misma área de la aplicación y que no tengan mucha complejidad (Tabares, 2010).

Este patrón está representado en las clases contenidas dentro del archivo View.py, las cuales tienen una única responsabilidad que puede ser CreateView, UpdateView o Delete.View

- Controlador: Le asigna a una clase la responsabilidad de manejar las respuestas del sistema a las acciones de los usuarios. El controlador define el método para la operación del sistema. Permite aumentar el potencial para la reutilización, asegura que la lógica de la aplicación no se maneje en la interfaz y permite validar las operaciones de manera eficiente y segura (Larman, 2003).

En la propuesta de solución el controlador se evidencia a través de una vista en Django y en la configuración de las rutas URL, permitiendo una asignación efectiva de responsabilidades en la gestión de solicitudes y lógica de la aplicación.

2. Pandilla de cuatro (GOF, por sus siglas en inglés): Son patrones usualmente aplicados en el trabajo orientado a objetos. Los mismos se clasifican de acuerdo a sus funciones en tres grupos: creacionales, estructurales y de comportamiento. De ellos se utiliza:

- Constructor: Separa la creación de un objeto de su representación, para que a través del mismo proceso de creación se obtengan distintas representaciones de un objeto. Tiene como ventajas reducir el acoplamiento y variar la representación interna de estructuras complejas, manteniendo una interfaz común desde el objeto fuente (Hunt, 2013).

Se emplea para la creación de objetos complejos desde un objeto fuente, el objeto fuente se compone de una variedad de partes que contribuyen individualmente a la creación de cada objeto complejo.

- Decorador: agrega funcionalidades adicionales a las vistas (ibíd.).

Se emplea este patrón para la creación de vistas personalizadas y *middleware*, permitiendo agregar funcionalidad adicional a las vistas o el procesamiento de solicitudes sin modificar el código principal como la autenticación.

2.9. Modelo de datos

Para la determinación de una estructura lógica de una base de datos y el modo de almacenar, organizar y manipular los datos se emplea el modelo de datos (Pressman, 2005). Con el objetivo de definir las clases persistentes se identifican los conceptos, en el dominio del negocio, que persisten en el tiempo. A partir del diagrama de clases persistentes se generó el siguiente diagrama entidad-relación, el cual posee un conjunto de tablas correspondientes a cada componente:

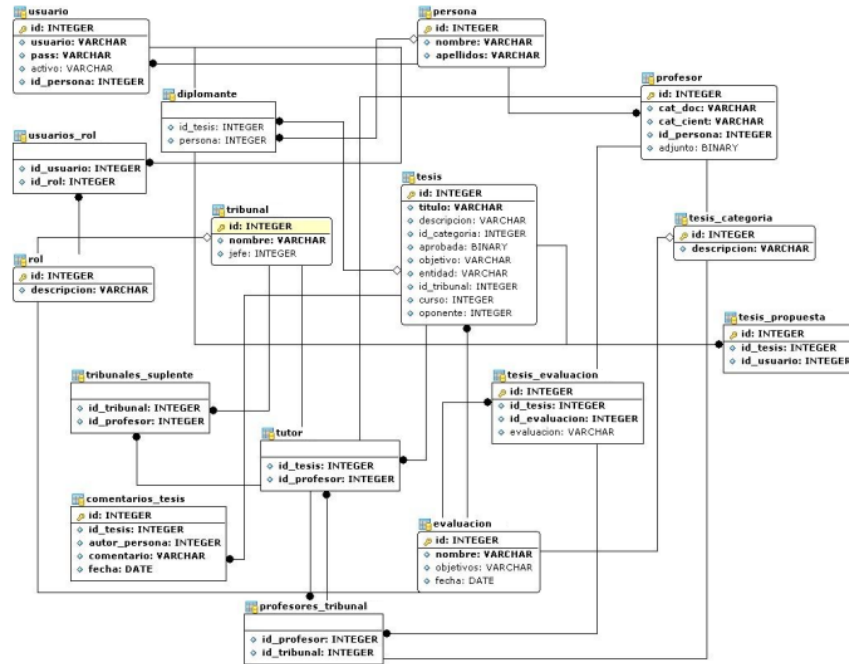


Figura 2.2. Modelo de datos

2.10. Consideraciones parciales

Luego de haber realizado el análisis y diseño del sistema propuesto y generar los artefactos que dispone la metodología seleccionada, se puede concluir lo siguiente:

- La descripción de la propuesta de solución a través del análisis de sus principales características, permitió identificar las funcionalidades del sistema propuesto, acorde a las necesidades del cliente para contribuir a la gestión de los trabajos de diploma de la Facultad 4 de la UCI.
- La especificación de los requisitos del sistema permitió describir cada uno de los requisitos funcionales a través de las HU, así como identificar los requisitos no funcionales teniendo en cuenta un conjunto de características generales y restricciones del mismo.
- El establecimiento de la arquitectura del sistema a partir del patrón arquitectónico, facilitó aislarse del problema en que se enmarca la presente investigación, asegurando un entendimiento común entre las partes interesadas.
- La identificación de los patrones de diseño permitió disminuir el impacto de los cambios futuros en el código fuente de la misma.
- Las características abordadas sobre el diseño del sistema a través de la comprensión de cada elemento que lo componen, facilitó el enfoque en cuanto a composición lógica y física de la propuesta de solución.

Implementación y pruebas del sistema para la gestión de los trabajos de diploma en la Facultad 4 de la UCI

En el presente capítulo se abordan los principales aspectos relacionados con la implementación de la solución propuesta. Se definen los estándares de codificación para lograr una mejor comprensión y estandarización del código resultante. Se realiza la verificación del correcto funcionamiento del software mediante la aplicación de las pruebas correspondientes.

3.1. Estándares de codificación

Las convenciones o estándares de codificación son pautas de programación que no están enfocadas a la lógica del programa, sino a su estructura y apariencia física para facilitar la lectura, comprensión y mantenimiento del código. Un estándar de codificación completo comprende todos los aspectos de la generación de código. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un código de alta calidad es de gran importancia para la calidad del software y para obtener un buen rendimiento.

A continuación, se define el estándar de codificación empleado en el sistema.

Tabla 3.1. Estándares de codificación a utilizar en la implementación del sistema. (Fuente: Elaboración propia).

Tipo de estándar	Descripción
Organización del código	• El código en una página se organizará por bloques • La indentación se realizará solamente con tabulaciones, no debe utilizarse nunca los cuatro (4) espacios
Continúa en la siguiente página	

Tipo de estándar	Descripción
Máxima longitud de las líneas	• Todas las líneas deben estar limitadas a un máximo de setenta y nueve caracteres • Dentro de paréntesis, corchetes o llaves se puede utilizar la continuación implícita para cortar las líneas largas.
Líneas en blanco	• Separar las funciones de alto nivel y definiciones de clases con dos (2) líneas en blanco • Las definiciones de métodos dentro de una clase deben separarse por una (1) línea en blanco • Se puede utilizar líneas en blanco escasamente para separar secciones lógicas
Importación	• Las importaciones deben estar en líneas separadas • Siempre deben colocarse al comienzo del archivo • Deben quedar agrupadas de la siguiente forma: ◦ Importaciones de la librería estándar ◦ Importaciones terceras relacionadas ◦ Importaciones locales de la aplicación / librerías • Cada grupo de importaciones debe estar separado por una línea en blanco
Continúa en la siguiente página	

Tipo de estándar	Descripción
Espacios en blanco en expresiones y sentencias	• Evitar utilizar espacios en blanco en las siguientes situaciones: ◦ Inmediatamente dentro de paréntesis, corchetes y llaves ◦ Inmediatamente antes de una coma, un punto y coma o dos puntos ◦ Inmediatamente antes del paréntesis que comienza la lista de argumentos en la llamada a una función ◦ Inmediatamente antes de un corchete que empieza una indexación ◦ Más de un espacio alrededor de un operador de asignación (u otro) para alinearlos con otro • Deben rodearse con exactamente un espacio los siguientes operadores binarios: ◦ Asignación (=) ◦ Asignación de aumentación (+=, -=, *=, /=) ◦ Comparación (==, <, >, >=, <=, =, <>, in, not in, is, is not) ◦ Expresiones lógicas (and, or, not) • Si se utilizan operadores con prioridad diferente se aconseja rodear con espacios a los operadores de menor prioridad • No utilizar espacios alrededor del igual (=) cuando es utilizado para indicar un argumento de una función o un parámetro con un valor por defecto
Comentarios	• Los comentarios deben ser oraciones completas • Si un comentario es una frase u oración su primera palabra debe comenzar con mayúscula a menos que sea un identificador que comience con minúscula • Si un comentario es corto el punto final puede omitirse • Los comentarios de una línea para aclaraciones del código aparecerán seguidos de los caracteres “//” en caso de código JavaScript mientras que en Python por el carácter “#” y deben ubicarse en la misma línea que se desea comentar
Continúa en la siguiente página	

Tipo de estándar	Descripción
Convenciones de nombramiento	• Nunca se deben utilizar como simples caracteres para nombres de variables los caracteres ele minúscula “l”, o mayúscula “O”, ele mayúscula “L” ya que en algunas fuentes son indistinguibles de los números uno (1) y cero (0) • Los módulos deben tener un nombre corto y en minúscula. • Los nombres de clases deben utilizar la convención “CapWords” (palabras que comienzan con mayúsculas) • Los nombres de las funciones deben estar escrito en minúscula separando las palabras con un guión bajo “_” • Las constantes deben quedar escritas con letras mayúsculas separando las palabras por un guión bajo (_)

A continuación, se muestra un fragmento de código donde se evidencia el uso de la codificación.

Código fuente 3.1. Ejemplo de código en Python

```

1 class TelegramRequestHandler(object):
2     def handle(self):
3         addr = self.client_address[0]          # Client IP-address
4         telegram = self.request.recv(1024)     # Recieve telegram
5         print "From: %s, Received: %s" % (addr, telegram)
6         return

```

3.2. Diagrama de despliegue

Los diagramas de despliegue muestran cómo los componentes de software se despliegan físicamente en los procesadores; es decir, el diagrama de despliegue muestra el hardware y el software en el sistema, así como el middleware usado para conectar los diferentes componentes en el sistema. En esencia, los diagramas de despliegue se pueden considerar como una forma de definir y documentar el entorno objetivo (Sommerville, 2011).

A continuación, la figura 3.1 muestra el diagrama correspondiente al sistema propuesto.

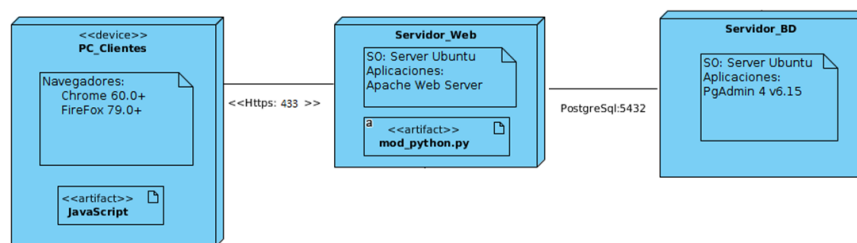


Figura 3.1. Representación del diagrama de despliegue.

PC_Clientes: Se refiere a el conjunto de todos los clientes que consumirán el software desde sus computadoras.

Servidor_Web: Elemento de cómputo, dedicado al procesamiento y la lógica de la ampliación web, alimentada por la aplicación servidora.

Servidor_BD: Elemento de cómputo, dedicado a almacenar y proveer datos necesarios para el funcionamiento de la aplicación web.

3.3. Tareas de ingeniería

La metodología XP, propone, que se describan las tareas de ingeniería necesarias para dar solución a cada HU. Las mismas, son descritas desde el punto de vista del programador, por lo que una HU puede tener más de una tarea de ingeniería asociada.

3.3.1. Tareas de ingeniería agrupadas por iteración

A continuación, la tabla 3.2 muestra las HU correspondientes a la iteración 1.

Tabla 3.2. Historias de usuario abordadas en la primera iteración (Fuente: Elaboración propia).

HU	Tiempo de estimación (semanas)	
	Estimación inicial	Estimación real
Autenticar usuario	0.5	0.5
Registrar usuario	0.2	0.2
Modificar usuario	0.5	0.5
Eliminar usuario	0.2	0.2
Listar usuario	0.2	0.2
Crear rol	0.5	0.5
Eliminar rol	0.5	0.5
Modificar rol	0.5	0.5
Crear perfil de tesis	0.5	0.5
Modificar perfil de tesis	0.5	0.5
Aprobar perfil de tesis	0.5	0.5
Mostrar detalles perfil de tesis	0.5	0.5
Eliminar perfil de tesis	0.5	0.5

A continuación, la tabla 3.3 muestra las HU correspondientes a la iteración 2.

Tabla 3.3. Historias de usuario abordadas en la segunda iteración (Fuente: Elaboración propia).

HU	Tiempo de estimación (semanas)	
	Estimación inicial	Estimación real
Crear problema investigativo	0.5	0.5

Eliminar problema investigativo	0.5	0.5
Modificar problema investigativo	0.5	0.5
Mostrar detalles problema investigativo	0.3	0.3
Convertir problema en perfil	0.2	0.2
Evaluar corte de tesis	0.5	0.5
Añadir las actas de los cortes de tesis	0.5	0.5
Descargar las actas de los cortes de tesis	0.2	0.2
Crear acta de predefensa	0.2	0.2
Eliminar acta de predefensa	0.5	0.5
Modificar acta de predefensa	0.5	0.5
Descargar de acta de predefensa	0.2	0.2
Mostrar detalles de acta de predefensa	0.2	0.2
Crear acta de defensa	0.2	0.2
Eliminar acta de defensa	0.2	0.2
Modificar acta de defensa	0.2	0.2
Descargar de acta de defensa	0.2	0.2
Mostrar detalles de acta de defensa	0.2	0.2

A continuación, la tabla 3.4 muestra las HU correspondientes a la iteración 3.

Tabla 3.4. Historias de usuario abordadas en la tercera iteración (Fuente: Elaboración propia).

HU	Tiempo de estimación (semanas)	
	Estimación inicial	Estimación real
Buscar perfil de tesis	0.5	0.5
Buscar problema investigativo	0.5	0.5
Buscar documento de tesis	0.2	0.2
Buscar usuario	0.2	0.2
Generar reporte de la evaluación del corte	0.2	0.2
Generar reporte de perfiles existentes	0.5	0.5
Generar reporte de perfiles aprobados	0.5	0.5
Generar reporte de perfiles sin aprobar	0.5	0.5
Generar reporte de perfiles en revisión	0.2	0.2
Generar reporte de tutores por Departamento	0.5	0.5
Generar reporte de estudiantes por Departamento	0.2	0.5
Adjuntar documento de tesis	0.2	0.2
Descargar documento de tesis	0.2	0.5
Mostrar documento de tesis	0.2	0.5
Crear tribunal de tesis	0.2	0.2
Eliminar tribunal de tesis	0.2	0.2
Modificar tribunal de tesis	0.5	0.5

Mostrar tribunal de tesis	0.2	0.2
Asignar estudiante a tribunal de tesis	0.5	0.5

Tabla 3.5. Tarea de ingeniería # 1

Tarea	
Número de tarea: 1	Número de Historia de usuario: 1
Nombre de la tarea: Autenticar usuario	
Tipo de tarea: Desarrollo	Puntos estimados: 0.5
Fecha de inicio: 5 de diciembre de 2022	Fecha de fin: 8 de diciembre de 2022
Programador responsable: Abel Machado Montes	
Descripción: Esta tarea permite que los usuarios se puedan autenticar en el sistema	

Tabla 3.6. Tarea de ingeniería # 2

Tarea	
Número de tarea: 2	Número de Historia de usuario: 7
Nombre de la tarea: Registrar usuario	
Tipo de tarea: Desarrollo	Puntos estimados: 0.2
Fecha de inicio: 9 de diciembre de 2022	Fecha de fin: 10 de diciembre de 2022
Programador responsable: Abel Machado Montes	
Descripción: Esta tarea permite registrar un usuario en el sistema	

Tabla 3.7. Tarea de ingeniería # 3

Tarea	
Número de tarea: 3	Número de Historia de usuario: 8
Nombre de la tarea: Modificar usuario	
Tipo de tarea: Desarrollo	Puntos estimados: 0.5
Fecha de inicio: 11 de diciembre de 2022	Fecha de fin: 14 de diciembre de 2022
Programador responsable: Abel Machado Montes	
Descripción: Esta tarea permite modificar un usuario registrado en el sistema	

Tabla 3.8. Tarea de ingeniería # 4

Tarea	
Número de tarea: 4	Número de Historia de usuario: 9
Nombre de la tarea: Listar usuario	
Tipo de tarea: Desarrollo	Puntos estimados: 0.2

Continúa en la próxima página

Tabla 3.8. Continuación de la página anterior

Fecha de inicio: 15 de diciembre de 2022	Fecha de fin: 16 de diciembre de 2022
Programador responsable: Abel Machado Montes	
Descripción: Esta tarea permite mostrar un listado con todos los usuarios registrado en el sistema	

Tabla 3.9. Tarea de ingeniería # 5

Tarea	
Número de tarea: 5	Número de Historia de usuario: 2
Nombre de la tarea: Crear perfil de tesis	
Tipo de tarea: Desarrollo	Puntos estimados: 0.5
Fecha de inicio: 20 de diciembre de 2022	Fecha de fin: 27 de diciembre de 2022
Programador responsable: Abel Machado Montes	
Descripción: Esta tarea permite registrar en el sistema un nuevo perfil de tesis	

3.4. Pruebas de software

Las pruebas intentan demostrar que un programa hace lo que se intenta que haga, así como descubrir defectos en el programa antes de usarlo. Al probar el software, se ejecuta un programa con datos artificiales. Hay que verificar los resultados de la prueba que se opera para buscar fallas, defectos, anomalías o información de requisitos no funcionales del programa (Sommerville, 2011).

A fin de encontrar fallas en el sistema y garantizar un nivel aceptable de calidad y confianza, se realizaron pruebas de software haciendo uso de las principales técnicas existentes, y aprovechando los módulos unittest.py y TestCase.py, ofrecidos por Python y Django respectivamente, que facilitan las pruebas automatizadas.

3.4.1. Pruebas de aceptación

Las pruebas de aceptación son pruebas formales, ejecutadas para verificar si un sistema satisface sus requerimientos de negocio. Estas pruebas requieren que el software se encuentre en funcionamiento, y se centran en replicar el comportamiento de los usuarios, a fin de rechazar cambios si no se cumplen los objetivos. Estos objetivos pueden ir más allá de obtener una respuesta específica (Pressman, 2005).

Para que este tipo de pruebas se lleve a cabo correctamente resulta importante se definan los criterios de aceptación justo antes de empezar a trabajar en el mismo. Así mismo, cualquier requerimiento adicional que surja durante el proceso deberá verse reflejado en tales criterios de aceptación.

Diseño de casos de prueba de aceptación

Las pruebas de aceptación fueron realizadas al culminar cada una de las iteraciones. A continuación, se muestran los casos de prueba de aceptación realizados para cada HU:

Tabla 3.10. Prueba de aceptación # 1

Caso de prueba de aceptación	
Código: HU1_P1	Historia de usuario: 1
Nombre: Autenticar usuario	
Descripción: Autenticar usuario en el sistema	
Condiciones de ejecución: El usuario debe estar registrado en la Base de Datos del sistema	
Pasos de ejecución: El usuario introduce los datos requeridos (usuario y contraseña) y posteriormente da clic en el botón Autenticar	
Resultados esperados: El usuario se autentica en el sistema	

Tabla 3.11. Prueba de aceptación # 2

Caso de prueba de aceptación	
Código: HU7_P1	Historia de usuario: 7
Nombre: Registrar usuario	
Descripción: Registrar usuario en el sistema	
Condiciones de ejecución: No aplica	
Pasos de ejecución: El usuario administrador registra un nuevo usuario	
Resultados esperados: El usuario añadido puedes hacer uso del sistema	

Tabla 3.12. Prueba de aceptación # 3

Caso de prueba de aceptación	
Código: HU9_P1	Historia de usuario: 9
Nombre: Listar usuario	
Descripción: Listar usuarios	
Condiciones de ejecución: Usuario autenticado	
Pasos de ejecución: EL usuario selecciona la opción de listar	

Continúa en la próxima página

Tabla 3.12. Continuación de la página anterior

Resultados esperados: El sistema muestra una lista con todos los usuarios registrados en la base de datos
--

Tabla 3.13. Prueba de aceptación # 4

Caso de prueba de aceptación	
Código: HU10_P1	Historia de usuario: 10
Nombre: Eliminar usuario	
Descripción: Eliminar usuario en el sistema	
Condiciones de ejecución: Usuario autenticado	
Pasos de ejecución: El usuario selecciona la opción de eliminar, luego aparece una alerta que muestra un mensaje si se desea eliminar el mensaje. En caso de que si se desee eliminar se presiona el botón aceptar, en caso contrario se presiona el botón cancelar.	
Resultados esperados: El usuario eliminado pasa a un estado inactivo en la base de datos	

Tabla 3.14. Prueba de aceptación # 5

Caso de prueba de aceptación	
Código: HU46_P1	Historia de usuario: 46
Nombre: Crear tribunal	
Descripción: Crear tribunal de tesis	
Condiciones de ejecución: Usuario autenticado con permisos de administrador	
Pasos de ejecución: El usuario selecciona la opción de crear tribunal y rellena los campos requeridos.	
Resultados esperados: Se crea un tribunal y se muestra un listado de los tribunales creados en el sistema.	

Tabla 3.15. Prueba de aceptación # 6

Caso de prueba de aceptación	
Código: HU50_P1	Historia de usuario: 50
Nombre: Asignar estudiante	
Descripción: Asignar estudiante a tribunal de tesis	

Continúa en la próxima página

Tabla 3.15. Continuación de la página anterior

Condiciones de ejecución: Usuario autenticado con permisos de administrador
Pasos de ejecución: El usuario selecciona la opción de asignar estudiante y marca o mediante un buscador elige los estudiantes que pertenecen al tribula, luego selecciona la opción de asignar y luego la opción de aceptar.
Resultados esperados: Se agregan los estudiantes al tribunal seleccionado

3.4.2. Prueba de seguridad

Las pruebas de seguridad se diseñan para sondear las vulnerabilidades del entorno lado cliente, las comunicaciones de red que ocurren conforme los datos pasan de cliente a servidor y viceversa, y el entorno del lado servidor. Cada uno de estos dominios puede atacarse, y es tarea del examinador de seguridad descubrir las debilidades que puedan explotar quienes tengan intención de hacerlo (Pressman, 2005).

Las pruebas de seguridad se aplicaron con ayuda de la herramienta Acunetix Web Vulnerability Scanner 9.5 que establece alertas de tipo: alta, media, baja e informacional, realizándose en dos iteraciones durante el desarrollo de la propuesta solución.

En una primera iteración se obtuvo un total de 22 alertas de seguridad, de las cuales 4 clasifican de nivel medio, 1 de nivel bajo y 17 informativas. De las de nivel medio, destacó el uso de protocolo no seguro para el envío de datos, así como los mensajes de error que se muestra en el modo DEBUG de Django para el desarrollo y se detectaron problemas para la protección de contra ataques de fuerza bruta en el formulario de autenticación.

La de nivel bajo, consistía en vistas del sitio que se podían acceder directamente sin pasar la autenticación y el sistema de roles establecido. De carácter informativo fueron detectadas posibles cuentas de usuario en ficheros, así como presencia de directorios desprotegidos y la existencia de etiquetas iframe de HTML5.

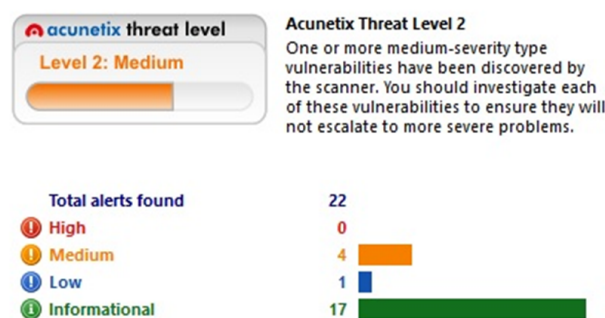


Figura 3.2. Prueba de seguridad 1ra iteración.

Después de aplicar refactorización del código y realizar las validaciones correspondientes, se aplicó la se-

gunda iteración en búsqueda de vulnerabilidades al sistema, arrojando como resultado que todas las que se habían detectado en la primera iteración, habían sido corregidas.

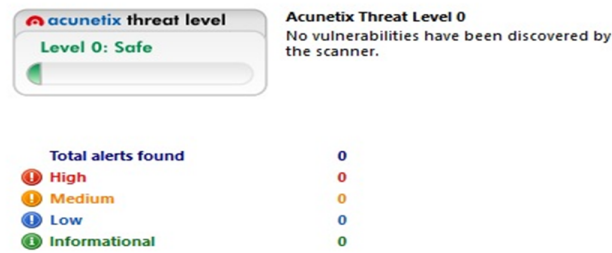


Figura 3.3. Prueba de seguridad 2da iteración.

3.4.3. Pruebas unitarias

Persiguiendo un correcto funcionamiento del código se emplea las pruebas unitarias (Pressman, 2005), utilizado para ello el método de caja blanca.

Método de caja blanca

El método tiene como objetivo principal probar el código. Entre las técnicas existentes de caja blanca se encuentran: prueba del camino básico, prueba de condición, prueba de flujo de datos y prueba de bucles. Para la investigación se utiliza la prueba del camino básico, la misma permite: generar el grafo de flujo, calcular la Complejidad Ciclomática, determinar los caminos linealmente independientes y diseñar los casos de prueba para forzar la ejecución de cada camino del conjunto básico.

Se utilizó el módulo TestCase que ofrece el framework, Django para la automatización de las pruebas a unidades de código. Se probó cada módulo desarrollado, y gracias a la aplicación de la técnica de camino básico se pudieron automatizar pruebas para cada uno de los escenarios o caminos posibles, garantizando probar todo el código en cuestión.

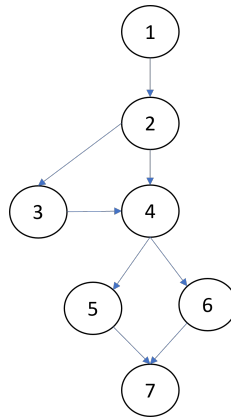
La tabla muestra técnica del camino básico que fue aplicada al método crear_Tribunal(), que se encarga de solicitar un nuevo tribunal en el sistema. Luego de la determinación de los nodos y flujos de control del código se obtuvo el grafo de flujo y se calculó la complejidad ciclomática del algoritmo según muestra en la tabla siguiente.

Tabla 3.16. Cálculo de la complejidad ciclomática del método crear_Tribunal().

Método

```
def crear_Tribunal(request):
    if request.method == "POST":
        form_nuevo = TribunalForm(request.POST, request.FILES)
        if form_nuevo.is_valid():
            form_nuevo.save()
            return redirect("tribunales")
        else:
            return render(request, "Tesis/formulario/formulario.html", {"form": form_nuevo})
```

Grafo resultante:



Complejidad Ciclomática:

$V(G) = \# \text{ de regiones}$

$V(G) = 3$

$$V(G) = A - N + 2$$

$$V(G) = 8 - 7 + 2$$

$$V(G) = 3$$

$$V(G) = P + 1$$

$$V(G) = 2 + 1$$

$$V(G) = 3$$

Como resultado se obtuvo que la complejidad ciclomática es igual a 3, lo que significa que existen 3 posibles caminos linealmente independientes y hay que diseñar un mínimo de 3 casos de prueba para el algoritmo crear_Tribunal(). La tabla 3.17 muestra los caminos existentes. En el anexo .5 se encuentran los casos de prueba diseñados.

Tabla 3.17. Caminos del grafo de flujo (Fuente: Elaboración propia).

No.	Camino
1	1,2,4,5,7
2	1,2,4,6,7
3	1,2,3,4,5,7

Con la realización de los casos de prueba diseñados se probó la ejecución de cada sentencia del código al menos una vez, teniendo en cuenta todas las condiciones lógicas en sus variantes verdaderas y falsas.

Los resultados del método de caja blanca fueron satisfactorios. La obtención de la complejidad ciclomática permitió determinar que el algoritmo es de baja complejidad.

3.4.4. Pruebas funcionales

Este tipo de prueba se realiza sobre el sistema funcionando, comprobando que cumpla con la especificación. Para estas pruebas, se utilizan las especificaciones de casos de prueba. Las pruebas basadas en requerimientos son pruebas de validación más que de defecto: se intenta demostrar que el sistema implementó adecuadamente sus requerimientos (Sommerville, 2011)

Método de caja negra

Las pruebas de caja negra, también llamadas pruebas de comportamiento, se enfocan en los requerimientos funcionales del software. Las técnicas de prueba de caja negra permiten derivar conjuntos de condiciones de entrada que revisarán los requerimientos funcionales para un programa (Pressman, 2005). El método de caja negra presenta varias técnicas de prueba como son: partición de equivalencia, análisis de valores límites y grafos de causa-efecto.

En la presente investigación se utilizará específicamente dentro del método de caja negra la técnica de partición de equivalencia, generando los casos de pruebas de dicha técnica sobre las diferentes interfaces que responden a los requisitos funcionales.

A continuación, la tabla 3.18 muestra el diseño de caso de pruebas del requisito “Autenticar usuario” donde se analizarán las variables y condiciones que puedan determinar la respuesta del sistema.

Tabla 3.18. Caso de prueba “Autenticar usuario” (Fuente: Elaboración Propia).

Escenario	Descripción	Variables		Respuesta esperada	Respuesta
		Usuario	Contraseña		
EC 1.1 Autenticar usuario correctamente	El usuario llena los campos y presiona el botón “Autenticar”.	V admin	V Admin123	El sistema autentica correctamente.	El sistema autentica correctamente.
EC 1.2 Autenticar usuario incorrectamente.	El usuario llena los campos con al menos un dato incorrecto y presiona el botón “Autenticar”.	V Admin	I Administrador	Se muestra un mensaje con la notificación “Credenciales Incorrectas” y se mantiene en el formulario de autenticación.	El sistema niega la autenticación.

EC 1.3 Autenticar usuario datos nulos	El usuario deja al menos un dato nulo y presiona el botón “Autenticar”.	I	V Admin123	Se muestra un mensaje con la notificación “Rellene este campo” y se mantiene en el formulario de autenticación.	El sistema niega la autenticación.
---------------------------------------	---	---	---------------	---	------------------------------------

Tabla 3.19. Variables de caso de prueba “Autenticar usuario” (Fuente: Elaboración Propia).

No.	Variable	Valor Nulo	Descripción
1	usuario	No	Es un campo que permite al usuario insertar su usuario correctamente. Acepta solo letras y no acepta espacio entre letras
2	Contraseña	No	Es un campo que permite al usuario insertar su contraseña correctamente. Acepta cualquier carácter alfa numérico.

En la tabla 3.20 también muestra el diseño de caso de pruebas del requisito “Registrar usuario” donde se analizarán las variables y condiciones que puedan determinar la respuesta del sistema.

Tabla 3.20. Caso de prueba “Registrar usuario” (Fuente: Elaboración Propia).

Escenario	Descripción	Variables				Respuesta esperada	Respuesta
		Usuario	Contraseña	Confirmar contraseña	Carnet		
EC 1.1. Insertar usuario correctamente.	El usuario llena los campos y presiona el botón “Guardar”.	V admi	V Adm.12	V Adm.12	V 98112399887	Se crea el usuario y se muestra un mensaje con la notificación “El usuario administrador fue agregado correctamente”.	El sistema inserta usuarios correctamente.

CAPÍTULO 3. IMPLEMENTACIÓN Y PRUEBAS DEL SISTEMA PARA LA GESTIÓN DE LOS TRABAJOS DE DIPLOMA EN LA FACULTAD 4 DE LA UCI

EC EC 1.2	El usuario llena los campos con al menos un dato incorrecto y presiona el botón “Guardar”.	V admi	V Adm.12	V Adm.12	I 981hjg	Se muestra un mensaje indicando el campo que posee datos incorrectos y se mantiene en el formulario.	El sistema niega la inserción del usuario.
EC 1.3	El usuario llena los campos con contraseñas diferentes y presiona el botón “Guardar”.	V admi	V Adm.12	I 12.Adm	V 98112399887	Se muestra un mensaje indicando que las contraseñas no coinciden y se mantiene en el formulario.	El sistema niega la inserción del usuario.
EC 1.4	El usuario deja al menos un dato nulo y presiona el botón “Guardar”.	I	V Adm.12	V Adm.12	I	Se muestra un mensaje indicando el campo obligatorio y se mantiene en el formulario.	El sistema niega la inserción del usuario.

Tabla 3.21. Variables de caso de prueba “Registrar usuario” (Fuente: Elaboración Propia).

No.	Variable	Valor Nulo	Descripción
1	Usuario	No	Es un campo que permite al usuario insertar su usuario correctamente. Acepta solo letras y no acepta espacio entre letras
2	Contraseña	No	Es un campo que permite al usuario insertar su contraseña en el formato adecuado. Acepta cualquier carácter alfa numérico.
3	Confirmación de la contraseña	No	Es un campo que permite al usuario confirmar su contraseña, la cual tiene que ser igual al campo anterior. Acepta cualquier carácter alfa numérico.
4	Carnet	No	Es un campo que permite al usuario insertar su carnet correctamente. Acepta solo letras números.

Las pruebas de caja negra se aplicaron con el objetivo de evaluar las interfaces de comunicación con el usuario, las que demostraron coherencia y funcionalidad, así como probar todas aquellas funcionalidades

directamente relacionadas con los requisitos funcionales del sistema. La técnica de partición de equivalencia es aplicada para evaluar los diferentes escenarios que pueden tener lugar ante la ejecución de una acción. Como resultado de la aplicación de estas pruebas se ejecutan las posibles variantes que posee una interfaz de comunicación con el usuario, resolviendo las no conformidades arrojadas y perfeccionando lo obtenido. Durante la realización de las pruebas se detectaron un conjunto de no conformidades relacionadas con errores de validación y funcionalidad. Los resultados se muestran en la figura 3.4, donde se evidencia la cantidad de casos de prueba ejecutados, los casos de prueba con no conformidades. Se realizaron cuatro iteraciones, durante la primera iteración se analizaron 50 casos de prueba, de los cuales 29 resultaron no conformidades. En la segunda y tercera iteración a través de las pruebas de regresión, con el uso del software Selenium IDE, se verificó que las no conformidades anteriores estuviesen solucionadas, y de estas pruebas se obtuvieron 18 y 13 nuevas no conformidades respectivamente, quedando resuelta en la cuarta iteración y cumpliéndose correctamente los requisitos funcionales.

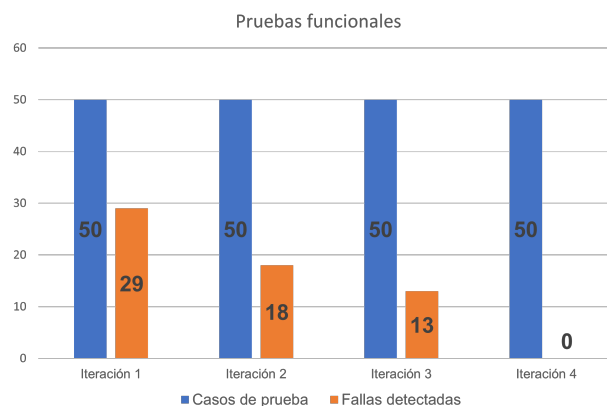


Figura 3.4. Representación del resultado de las pruebas.

3.4.5. Prueba de rendimiento

Las pruebas de rendimiento deben diseñarse para garantizar que el sistema procese su carga pretendida. Esto implica efectuar una serie de pruebas donde se aumenta la carga, hasta que el rendimiento del sistema se vuelve inaceptable. Las pruebas de rendimiento se preocupan tanto por demostrar que el sistema cumple con sus requerimientos, como por descubrir problemas y defectos en el sistema (Sommerville, 2011).

La prueba de carga y estrés se refiere, generalmente, a la práctica de comprobar el comportamiento de una aplicación mediante cargas o entradas pesadas. Las mismas se realizan con el fin de verificar si el sistema satisface los requisitos de rendimiento para situaciones críticas como pueden ser: la cantidad límite de usuarios accediendo de forma concurrente a los servicios brindados, documentos extremadamente grandes, cantidad de transacciones que se pueden procesar de forma concurrente cada minuto, tiempo de respuesta, entre otros (Fernández Buchillón, 2019).

Para aplicar estas pruebas se tomó como muestra una cantidad de 400 usuarios. Las pruebas se desarrollaron

con el apoyo de la herramienta Apache JMeter 5.5, simulando el entorno de producción para obtener los datos más cercanos al comportamiento y resultado real.

El entorno elegido cuenta con las siguientes características:

Hardware:

- **Microprocesador:** Intel Core (TM) i5 7200U CPU 2.40 GHz
- **Memoria:** Memoria de acceso aleatorio (RAM, por sus siglas en inglés): 12GB
- **Tarjeta de Red:** Ethernet 10/100 Mbps

Software:

- Sistema Operativo: Windows 10, Arquitectura de 64 bits

Para las pruebas de carga se tuvo en cuenta la cantidad de usuarios en la universidad seleccionando como mínimo 100 usuarios concurrentes. Como se muestra en la figura 3.5, se obtuvo un tiempo promedio de respuesta del servidor de 2 segundo cumpliendo satisfactoriamente dicho requisito no funcional

Etiqueta	# Muestras	Media	Min	Máx	Desv. Estándar	%Error	Rendimien...	Kb/sec	Sent KB/sec	Media de Eyt...
Petición HTTP	400	11915	686	25474	7362,74	0,67%	2,3/sec	108,64	0,67	29603,5
Total	400	11915	686	25474	7362,74	0,67%	2,3/sec	108,64	0,67	29603,5

Figura 3.5. Resultado de pruebas de rendimiento

En prueba de estrés como se muestra en la figura 3.6, se obtuvo para 400 usuarios concurrentes un tiempo de respuesta promedio mayor que 5 segundos superando el tiempo establecido.

Etiqueta	# Muestras	Media	Min	Máx	Desv. Está...	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de ...
Petición H...	400	8215	319	19102	6062,03	31,50%	7,3/sec	87,40	0,30	38581,9
Total	400	8215	319	19102	6062,03	31,50%	7,3/sec	87,40	0,30	38581,9

Figura 3.6. Resultado de pruebas de rendimiento

Con la aplicación de las pruebas de rendimiento se arrojó que el sistema responde como se esperaba. Además, la realización de una prueba de estrés complementaria, permitió saber el momento justo en el que el sistema empieza a fallar y a obtener tiempos de respuesta inaceptables para una correcta experiencia de usuario, superiores a los 5 segundos.

3.5. Consideraciones parciales

En este capítulo se abordaron los elementos del sistema para la gestión de la información de los trabajos de diplomas, así como las pruebas realizadas al mismo y los resultados obtenidos; lo que permite concluir:

- La implementación de la propuesta de solución facilitó obtener una aplicación funcional capaz de contribuir con la gestión de la información de los trabajos de diplomas de la Facultad 4 de la UCI.

- La validación de la propuesta de solución, mediante las pruebas de software facilitó corroborar la calidad de la misma, estableciendo así el cumplimiento de los requisitos definidos por el cliente.

El presente Trabajo de Diploma concluye con el desarrollo de un Sistema para la gestión de la información de los trabajos de diplomas de la Facultad 4 en la UCI.

- El análisis y la fundamentación teórica de los principales conceptos asociados a los trabajos de diplomas, permitió comprender el alcance y relevancia de la investigación.
- El estudio y análisis de la gestión de los trabajos de diplomas dentro de la UCI, permitió conocer en qué estado se encontraba, y corroborar la necesidad de la creación de una solución que contribuyera a la ejecución de dicho proceso.
- El diseño del sistema informático para la gestión de la información de los trabajos de diplomas, sentó las bases para la implementación del mismo, lo cual no hubiese sido posible sin el análisis que se realizó de soluciones similares que permitieron detectar más requisitos al sistema, en conjunto con los extraídos durante las entrevistas con los clientes.
- Con la implementación del sistema, se creó un producto de software, sobre la base de la tecnología web, que permitió contribuir a la gestión de la información asociada a los trabajos de diplomas.
- La aplicación de las pruebas, permitió comprobar el correcto funcionamiento del Sistema para la Gestión de la información de los trabajos de diplomas, a partir del cumplimiento de los requisitos pactados con el cliente.

Recomendaciones

Para dar continuidad a la presente investigación se recomienda:

- Integrar el sistema con los la Base de Datos de la UCI para obtener los datos reales de los estudiantes y profesores, así como las áreas de la universidad.
- Adicionarle nuevas funcionalidades con el objetivo de permitir envíos de notificaciones por correo electrónico y el vínculo con el sistema Akademos.

Akademios Sistema de Gestión Académica. 10, 11

CASE Ingeniería de Software Asistida por Computadora. 13

CRC Clase-Responsabilidad-Colaboración. 30

CSS Hoja de estilo en Cascada. 15, 17, 32

CSS3 Hoja de estilo en Cascada Versión 3. 15, 22

CT Comité o Comisión de Tesis. 2

DRY No te repitas. 16

GOF Pandilla de cuatro. 34

GRASP Patrones Generales de Software de Asignación de Responsabilidad. 33

HTML Leguaje de marcado de hipertexto. 15

HTML5 Leguaje de marcado de hipertexto Versión 5. 15, 17, 22, 46

HU Historias de Usuario. 22, 26–30, 35, 40, 41, 44

IDE Entorno de Desarrollo Integrado. 15

IES Instituciones de Educación Superior. 1

LERU Liga de Universidades Europeas de Investigación. 1

MES Ministerio de Educación Superior. 1, 2

MTV Modelo Vista Plantilla. 15, 32

MVC Modelo Vista Controlador. 15, 32

ORM Mapeador de Objeto Relacional. 16

PBS Public Broadcasting System. 16

POO Programación Orientada a Objeto. 13, 14

RAM Memoria de acceso aleatorio. 53

REPXOS 4.0 Repositorio Digital. 10, 11

SGBD Sistema Gestor de Base de Datos. 17

SGML El Lenguaje de marcado generalizado estándar. 15
SIDEP Sistema Informático de Estudios de Postgrado. 8, 11
SIGENU Sistema de Gestión de la Nueva Universidad. 9, 11
SQL Lenguaje de Consultas Estructurado. 16

UCI Universidad de las Ciencias Informáticas. 1–56, 62
UI Interfaz de Usuario. 21
UML Lenguaje de Modelado Unificado. 13, 22

W3C El Consorcio de la Red Informática Mundial. 15
WWW Red Informática Mundial. 15

XP Programación Extrema. 13, 18, 22, 40

Referencias bibliográficas

- BALOGUN, MO, 2022. Comparative Analysis of Complexity of C++ and Python Programming Languages. *Asian J. Soc. Sci. Manag. Technol.* Vol. 4, págs. 1-12 (vid. pág. 14).
- BARZAGA-SABLÓN, Oscar Santiago; PINCAY, Hugo Jesús Juan Vélez; NEVÁREZ-BARBERÁN, José Víctor y COBEÑA, María Verónica Arroyo, 2019. Gestión de la información y toma de decisiones en organizaciones educativas. *Revista de ciencias sociales.* Vol. 25, n.º 2, págs. 120-130 (vid. pág. 7).
- BOULTON, Geoffrey y LUCAS, Colin, 2011. What are universities for? *Chinese Science Bulletin.* Vol. 56, n.º 23, págs. 2506-2517 (vid. pág. 1).
- CENTELLAS COARITE, Melissa, 2015. *Sistema de control y gestión de historiales clínicos apoyado en dispositivos móviles Caso: Centro Medico La Paz.* Tesis doctoral (vid. pág. 26).
- Como hacer una tesis*, 1994 (vid. pág. 5).
- CONSORTIUM, World Wide Web et al., 1999. HTML 4.01 specification (vid. pág. 15).
- COORDINACIÓN UNIVERSITARIA, Consejo de, 2005. TESEO-Base de datos de Tesis Doctorales (vid. pág. 8).
- COSTA, Grisel Infante, 2007. Akademos, un Sistema Automatizado para la Gestión Académica. *Serie Científica de la Universidad de las Ciencias Informáticas.* Vol. 1, n.º 1 (vid. pág. 10).
- CUBA (PCC), Partido Comunista de, 2016. *Plan Nacional de desarrollo económico y social hasta el 2030: Propuesta de visión de la nación, ejes y sectores estratégicos. Aprobado por el VII Congreso del PCC.* Autor La Habana (vid. pág. 1).
- CUJAE, 2022. *Sigenu - Portal Docente* [online] [visitado 2022-06-08]. Disponible desde: <https://sigenu.cujae.edu.cu/sigenu-portal/#modues> (vid. pág. 9).
- DALLOS BUSTOS, Liliana Patricia; MONCADA CASTAÑO, Daniel Steven; ARIZA HERNÁNDEZ, Diana Carolina; FRANCO VARGAS, Verónica Johana et al., 2020. *Análisis comparativo entre metodologías ágiles y tradicionales para la gerencia de proyectos.* B.S. thesis. Universidad EAN.
- EDUCACIÓN SUPERIOR, Ministerio de, 2022. *Reglamento para el trabajo docente metodológico en la Educación Superior.* Ministerio de Justicia La Habana (vid. págs. 2, 6).
- FERNÁNDEZ BUCHILLÓN, Luis Ernesto, 2019. *Sistema para la gestión de la prenomina en el Centro de Ideoinformática.* B.S. thesis. Universidad de las Ciencias Informáticas. Facultad 1. (vid. pág. 52).

- FONSECA ARISTIGÜI, Liliana Victoria y SANABRIA BALBER, Lisdairy, 2017. *ProFeatPred: Sistema experto basado en modelos de rasgos de contacto bilineales para la predicción de propiedades biológicas en proteínas*. B.S. thesis. Universidad de las Ciencias Informáticas. Facultad 6. (vid. pág. 13).
- GAMMA, Erich; HELM, Richard; JOHNSON, Ralph; JOHNSON, Ralph E; VLISSIDES, John et al., 1995. *Design patterns: elements of reusable object-oriented software*. Pearson Deutschland GmbH (vid. pág. 33).
- GÓMEZ BERMEJO, Rafael, 2018. Aplicación web para la gestión y búsqueda de eventos (vid. pág. 17).
- HUNT, John, 2013. Gang of four design patterns. En: *Gang of four design patterns. Scala design patterns*. Springer, págs. 135-136 (vid. pág. 34).
- JACOB THORNTON, Mark Otto, 2022. *Get started with Bootstrap* [online] [visitado 2022-06-27]. Disponible desde: <https://getbootstrap.com/docs/5.2/getting-started/introduction/> (vid. pág. 17).
- JESÚS DOMÍNGUEZ, José de y LEGAL, Torres Representante. ORIENTACION Y LINEAMIENTOS GENERALES PARA ASESORAR LA INVESTIGACION, ELABORACION Y PRESENTACIÓN DE LA TESIS RECEPCIONAL (vid. pág. 5).
- JOVENCLUB, Revista, 2020. *Gestion de información*. Url: <https://revista.jovenclub.cu/gestion-de-informacion/> (vid. pág. 7).
- LABRADA DELGADO, Eduardo José, 2019. *Módulo Referencias Bibliográficas de XABAL REPXOS 4.0 en la tecnología Alfresco Community Edition 5.2*. B.S. thesis. Universidad de las Ciencias Informáticas. Facultad 2. (vid. pág. 11).
- LARA, Cecilia y FIGUEROA, Liliana Maria, 2020. Metodología ágil para el desarrollo de aplicaciones móviles educativas. En: *Metodología ágil para el desarrollo de aplicaciones móviles educativas. XV Congreso Nacional de Tecnología en Educación y Educación en Tecnología (TE&ET 2020)(Neuquén, 6 y 7 de julio de 2020)* (vid. pág. 30).
- LARMAN, Craig, 2003. UML y Patrones (vid. pág. 33).
- LUJÁN MORA, Sergio, 2016. *Programación de aplicaciones web. Historia. Principios básicos y clientes web*. Luján Mora, Sergio (vid. pág. 15).
- MAIDA, Esteban Gabriel y PACIENZIA, Julián, 2015. Metodologías de desarrollo de software (vid. pág. 12).
- MALQUI, Diego David, 2018. Diseño y desarrollo Web accesible utilizando HTML5 y CSS3 con nivel de conformidad A, respecto a las pautas WCAG (vid. pág. 15).
- MES, 2022. *Ministerio | MES* [online] [visitado 2022-04-12]. Disponible desde: <https://www.mes.gob.cu/ministerio> (vid. pág. 1).
- MICROSOFT, 2022. *Get Started with Visual Studio Code* [online] [visitado 2022-06-08]. Disponible desde: <https://code.visualstudio.com/learn/overview> (vid. pág. 15).

- MONTERO, Bryan Molina; CEVALLOS, Harry Vite y CUESTA, Jefferson Dávila, 2018. Metodologías ágiles frente a las tradicionales en el proceso de desarrollo de software. *Espirales revista multidisciplinaria de investigación*. Vol. 2, n.º 17, págs. 114-121 (vid. pág. 12).
- NISKA, Christoffer, 2014. *Extending Bootstrap*. Packt Publishing (vid. pág. 17).
- PARADIGM, Visual, 2022. *About Visual Paradigm* [online] [visitado 2022-06-24]. Disponible desde: <https://www.visual-paradigm.com/aboutus/> (vid. pág. 13).
- PÉREZ, Javier Eguíluz, 2019. *introduccion a JavaScript* (vid. pág. 14).
- PÉREZ, Maidelyn Díaz, 2017. Sistemas de gestión de información y conocimiento en empresas cooperativas: sociedades colaborativas de conocimiento. *Cooperativismo y Desarrollo: COODES*. Vol. 5, n.º 2, págs. 221-232 (vid. pág. 7).
- PILICITA GARRIDO, Anabel; BORJA LÓPEZ, Yolanda y GUTIÉRREZ CONSTANTE, Gonzalo, 2021. Rendimiento de MariaDB y PostgreSQL (vid. pág. 17).
- PRESSMAN, Roger S, 2005. *Software engineering: a practitioner's approach*. Palgrave macmillan (vid. págs. 13, 33, 34, 43, 46, 47, 49).
- PUIG, Jordi Collell, 2013. CSS3 y Javascript avanzado. *Diponible en <https://openlibra.com/es/book/download/css3-y-javascript-avanzado>* (vid. pág. 15).
- RAJME, Claudia Aguilar; GUERRA, Gabriel Alberto Pérez y GARCÍA, Neili Machado. SISTEMA PARA LA GESTIÓN DE PROCESOS DE EDUCACIÓN A DISTANCIA EN LAS SECRETARÍAS DOCENTES SYSTEM FOR THE MANAGEMENT OF DISTANCE EDUCATION PROCESSES IN THE EDUCATION SECRETARIATS (vid. pág. 9).
- RUBIO, Daniel, 2017. *Beginning Django*. Springer (vid. pág. 15).
- SALEH, Sabbir M; HUQ, Syed Maruful y RAHMAN, M Ashikur, 2019. Comparative study within Scrum, Kanban, XP focused on their practices. En: *Comparative study within Scrum, Kanban, XP focused on their practices. 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, págs. 1-6 (vid. pág. 13).
- SANGAMA OÑATE, Abel Fernando, 2020. Metodologías ágiles Scrum, XP, SLeSS, Scrumban, HME, Mobile-D y MASAN empleadas en la industria de dispositivos móviles: Un contraste en favor de la industria del desarrollo móvil (vid. págs. 12, 13).
- AL-SAQQA, Samar; SAWALHA, Samer y ABDELNABI, Hiba, 2020. Agile Software Development: Methodologies and Trends. *International Journal of Interactive Mobile Technologies*. Vol. 14, n.º 11 (vid. pág. 12).
- SOMMERVILLE, Ian, 2011. *Ingeniería de software* [online] [visitado 2022-06-18]. ISBN 978-607-32-0604-4 978-607-32-0603-7. Disponible desde: http://www.ingebook.com/ib/NPcd/IB_Escritorio_Visualizar?cod_primaria=1000193&libro=1518. OCLC: 1257013649 (vid. págs. 31, 39, 43, 49, 52).

- SRINATH, KR, 2017. Python—the fastest growing programming language. *International Research Journal of Engineering and Technology (IRJET)*. Vol. 4, n.º 12, págs. 354-357 (vid. pág. 14).
- TABARES, Ricardo Botero, 2010. Patrones grasp y anti-patrones: un enfoque orientado a objetos desde logica de programacion. *Entre Ciencia e Ingeniería*. Vol. 4, n.º 8, págs. 161-173 (vid. pág. 33).
- UCI, 2022. xauce-akademos-10. Url: <https://www.uci.cu/investigacion-y-desarrollo/productos/xauce/xauce-akademos-10> (vid. pág. 10).
- VAN ROSSUM, Guido et al., 2007. Python Programming language. En: *Python Programming language. USENIX annual technical conference*. Vol. 41, págs. 1-36. N.º 1 (vid. pág. 13).
- VEGA ALMEIDA, RL y ESTUDIO, Sánchez Ladrón de Guevara S, 2001. informétrico de la colección: trabajos de diploma de la especialidad Bibliotecología y Ciencia de la Información: período 1995-2000 [Trabajo de Diploma]. *La Habana: Universidad de La Habana* (vid. pág. 5).
- VENEZUELA, Universidad Central de, 2022. Url: <http://www.postgrado.ucv.ve/> (vid. pág. 8).
- VENTURA BAUTISTA, Jesús Natividad, 2021. JQUERY, AJAX Definición de jQuery, programar JavaScript con jQuery. Selección de un elemento del documento mediante el id. Selección de elementos por el tipo de elementos. Método text (), text (valor). Métodos attr (nombre de propiedad), attr (nombre de propiedad, valor) y removeAttr (nombre de propiedad). Métodos addClass y removeClass. Método html () y html (valor). Administración de eventos con jQuery. Eventos mouseover y mouseout. Tipos de eventos, efectos, tipos efectos, iteración, Ajax, funciones anónimas, ejemplos y aplicaciones (vid. pág. 17).
- VIDAL-SILVA, Cristian L; SÁNCHEZ-ORTIZ, Aurora; SERRANO, Jorge y RUBIO, José M, 2021. Experiencia académica en desarrollo rápido de sistemas de información web con Python y Django. *Formación universitaria*. Vol. 14, n.º 5, págs. 85-94 (vid. pág. 16).
- WIRFS-BROCK, Allen y EICH, Brendan, 2020. JavaScript: the first 20 years. *Proceedings of the ACM on Programming Languages*. Vol. 4, n.º HOPL, págs. 1-189 (vid. pág. 15).

Apéndice

.1. Modelo de entrevista aplicada a profesores

1. Sobre el proceso del ejercicio para la culminación de estudio que se lleva a cabo en la universidad, ¿considera usted que se pueda llevar a cabo la informatización del mismo?
2. ¿Qué debería de tener un sistema informático para que abarque el proceso y sea lo más ameno posible?
3. ¿Utilizaría usted la aplicación propuesta a la hora de gestionar el proceso en la UCI?
4. ¿Cómo se desarrolla el proceso en la facultad? ¿Qué herramientas utilizan para el apoyo y organización de este?
5. ¿Considera usted que el sistema que se propone permitiría una correcta gestión del proceso?

.2. Guía de observación para el análisis de los sistemas similares

Tabla 22. Guía de observación para estudio de soluciones (Fuente: Elaboración propia).

No.	Indicadores	D	R	B	MB	E
1	Gestionan los trabajos de diploma.					
2	Permiten la creación de tribunales de tesis.					
3	Permiten asignar estudiantes a tribunales de tesis.					
4	Permiten visualizar, cargar y descargar los documentos de tesis.					
5	Permiten dar seguimiento a los talleres de tesis.					

Leyenda

- Deficiente
- Regular
- Bien
- Muy bien
- Excelente

.3. Historias de usuario

Tabla 23. Historia de usuario # 6

Historia de usuario	
Número: 6	Nombre: Autenticar usuario
Usuario: Usuario	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta

Continúa en la próxima página

Tabla 23. Continuación de la página anterior

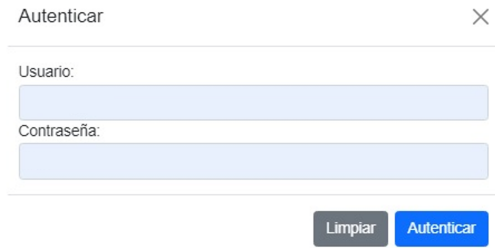
Puntos estimados: 1	Iteración asignada: 1
Programador responsable: Abel Machado Montes	
Descripción: El sistema debe proporcionar que el usuario pueda autenticarse, insertando, su usuario y contraseña, validando que no deje campos vacíos y que los mismos sean correctos, luego de su autenticación el sistema debe re direccionar al usuario hacia la pantalla inicial.	
Observaciones: El sistema debe mostrar una notificación si el usuario y contraseña son incorrectos.	
Interfaz: 	

Tabla 24. Historia de usuario # 7

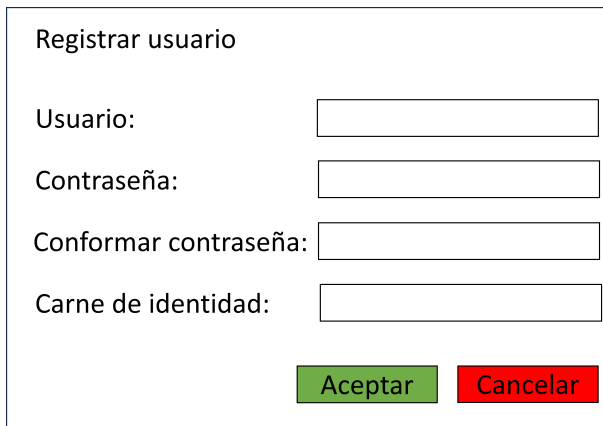
Historia de usuario	
Número: 7	Nombre: Registrar usuario
Usuario: Administrador	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 2
Programador responsable: Abel Machado Montes	
Descripción: El sistema debe ser capaz de agregar un nuevo usuario.	
Observaciones: Este proceso solo lo puede realizar el administrador.	
Interfaz: 	

Tabla 25. Historia de usuario # 8

Historia de usuario	
Número: 8	Nombre: Modificar usuario
Usuario: Administrador	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 3
Programador responsable: Abel Machado Montes	
Descripción: El sistema debe ser capaz de modificar un usuario seleccionado por el cliente.	
Observaciones: Este proceso solo lo puede realizar el administrador.	
Interfaz: Modificar usuario Usuario: Contraseña: Conformar contraseña: Carne de identidad: Aceptar Cancelar	

Tabla 26. Historia de usuario # 9

Historia de usuario	
Número: 9	Nombre: Eliminar usuario
Usuario: Administrador	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 4
Programador responsable: Abel Machado Montes	
Descripción: El sistema debe ser capaz de eliminar un usuario seleccionado por el cliente. El sistema muestra un mensaje de advertencia “Desea eliminar el elemento seleccionado”, y dos botones “Aceptar” y “Cancelar”	
Observaciones: Este proceso solo lo puede realizar el administrador.	

Continúa en la próxima página

Tabla 26. Continuación de la página anterior

Interfaz:

“Desea eliminar el elemento seleccionado”

Aceptar
Cancelar

Tabla 27. Historia de usuario # 10

Historia de usuario	
Número: 10	Nombre: Mostrar detalles perfil de tesis
Usuario: Todos los usuarios	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 12
Programador responsable: Abel Machado Montes	
Descripción: El sistema debe permitir mostrar todos los datos de un perfil de tesis, a fin de obtener su información de forma más organizada.	
Observaciones: Este proceso lo puede hacer cualquier usuario.	
Interfaz: Detalles de Perfil de tesis Autor: Abel Machado Problema de investigación: Objetivo general: Objetivos específicos : Tarea de investigación:  Documento.pdf Aceptar Cancelar	

Tabla 28. Historia de usuario # 11

Historia de usuario	
Número: 11	Nombre: Mostrar tribunal de tesis

Continúa en la próxima página

Tabla 28. Continuación de la página anterior

Usuario: Todos los usuarios													
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta												
Puntos estimados: 1	Iteración asignada: 49												
Programador responsable: Abel Machado Montes													
Descripción: El sistema debe permitir mostrar todos los datos de un tribunal de tesis, a fin de obtener su información de forma más organizada.													
Observaciones: Este proceso lo puede hacer cualquier usuario.													
Interfaz: Detalles de Tribunal de tesis Presidente: Miembro: Miembro: Miembro : Estudiantes <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: black; color: white;"> <th>Nombre y Apellidos</th><th>Tema de Tesis</th></tr> </thead> <tbody> <tr><td> </td><td> </td></tr> <tr><td> </td><td> </td></tr> <tr><td> </td><td> </td></tr> <tr><td> </td><td> </td></tr> <tr><td> </td><td> </td></tr> </tbody> </table> Aceptar Cancelar		Nombre y Apellidos	Tema de Tesis										
Nombre y Apellidos	Tema de Tesis												

Tabla 29. Historia de usuario # 12

Historia de usuario	
Número: 12	Nombre: Listar usuario
Usuario: Administrador	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 8
Programador responsable: Abel Machado Montes	
Descripción: El sistema debe permitir listar todos los usuarios registrados. Y permite la opción de modificar o eliminar algún usuario.	
Observaciones: usuario autenticado	

Continúa en la próxima página

Tabla 29. Continuación de la página anterior

Interfaz:

Usuarios

Nombre y apellido	Usuario	Departamento	Acciones
Abel Machado	abelM	Informatica	✎ 🗑
Abel Roque	aroque	Inteligencia Computacional	✎ 🗑
Pedro Juan Martínez	pjmartinez	Informática	✎ 🗑

Aceptar
Cancelar

Tabla 30. Historia de usuario # 13

Historia de usuario																	
Número: 13	Nombre: Aprobar perfil de tesis																
Usuario: profesor																	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta																
Puntos estimados: 1	Iteración asignada: 11																
Programador responsable: Abel Machado Monteso																	
Descripción: El sistema debe permitirle a un profesor aprobar un perfil de tesis. El profesor selecciona el perfil que desea aprobar y tiene la opción de aprobarlo dando clic en el botón “Aprobar”.																	
Observaciones: Usuario autenticado																	
Interfaz: Perfiles de tesis Buscar <input style="width: 15px; height: 15px; border: 1px solid black; border-radius: 50%; text-align: center; line-height: 15px;"/> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: black; color: white;"> <th>Autor</th> <th>Nombre de la tesis</th> <th>Tutores</th> <th>Acciones</th> </tr> </thead> <tbody> <tr> <td>Abel Machado</td> <td>Sistema para la gestión de las tesis en la Facultad 4</td> <td>Mario Ramírez</td> <td style="text-align: center;"> Aprobar </td> </tr> <tr> <td> </td> <td> </td> <td> </td> <td> </td> </tr> <tr> <td> </td> <td> </td> <td> </td> <td> </td> </tr> </tbody> </table> Aceptar Cancelar		Autor	Nombre de la tesis	Tutores	Acciones	Abel Machado	Sistema para la gestión de las tesis en la Facultad 4	Mario Ramírez	Aprobar								
Autor	Nombre de la tesis	Tutores	Acciones														
Abel Machado	Sistema para la gestión de las tesis en la Facultad 4	Mario Ramírez	Aprobar														

Tabla 31. Historia de usuario # 14

Historia de usuario	
Número: 14	Nombre: Crear tribunal de tesis
Usuario: Administrador, profesor	

Continúa en la próxima página

Tabla 31. Continuación de la página anterior

Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 46
Programador responsable: Abel Machado Monteso	
Descripción: El sistema debe permitir un crear un tribunal de tesis. Para ello el usuario debe seleccionar de los profesores existentes en base de datos quien tiene el rol de presidente, el de secretario y el de vocal.	
Observaciones: Usuario autenticado y con los permisos necesarios, y los profesores deben estar registrados en el sistema	
Interfaz: Crear de Tribunal de tesis Identificador: Presidente: Elige ▼ Secretario: Elige ▼ Vocal: Elige ▼ Aceptar Cancelar	

Tabla 32. Historia de usuario # 15

Historia de usuario	
Número: 15	Nombre: Modificar tribunal de tesis
Usuario: Profesor,administrador	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 48
Programador responsable: Abel Machado Monteso	
Descripción: El sistema debe permitir un modificar un tribunal de tesis. Para ello el usuario puede modificar los profesores del tribunal, así como la fecha y el identificador.	
Observaciones: Usuario autenticado y con los permisos necesarios, y los profesores deben estar registrados en el sistema	

Continúa en la próxima página

Tabla 32. Continuación de la página anterior

Interfaz:

Modificar Tribunal de tesis

Identificador:

Presidente:

Secretario:

Vocal:

Tabla 33. Historia de usuario # 16

Historia de usuario	
Número: 16	Nombre: Eliminar tribunal de tesis
Usuario: Profesor,administrador	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 47
Programador responsable: Abel Machado Monteso	
Descripción: El sistema debe permitir un eliminar un tribunal de tesis. Luego que el usuario seleccione la opción de eliminar, aparecerá una pantalla mostrando el mensaje “Desea eliminar el elemento seleccionado” y dos botones uno de “Aceptar” y uno de “Cancelar”.	
Observaciones: Usuario autenticado y con los permisos necesarios, y los profesores deben estar registrados en el sistema	
Interfaz: “Desea eliminar el elemento seleccionado” Aceptar Cancelar	

Tabla 34. Historia de usuario # 17

Historia de usuario	
Número: 17	Nombre: Crear problema investigativo
Usuario: Todos los usuarios	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 14
Programador responsable: Abel Machado Monteso	
Descripción: El sistema permite al usuario poder acceder a la opción de registrar el problema a dar solución, donde se muestra una página con un formulario a rellenar, con los campos siguientes: • Nombre • Área a la que tributa el problema • Línea de investigación a la que tributa el problema	
Observaciones: usuario autenticado	
Interfaz: Crear problema investigativo Nombre: Área: Línea de investigación: Problema: Aceptar Cancelar	

Tabla 35. Historia de usuario # 18

Historia de usuario	
Número: 18	Nombre: Eliminar problema investigativo
Usuario: administrador, profesor	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 15
Programador responsable: Abel Machado Monteso	
Descripción: El sistema permite eliminar un problema registrado en la base de datos. Mostrando un mensaje de advertencia “Desea eliminar el elemento seleccionado” y dos botones “Aceptar” y “Cancelar”.	

Continúa en la próxima página

Tabla 35. Continuación de la página anterior

Observaciones: usuario autenticado
Interfaz: “Desea eliminar el elemento seleccionado” Aceptar Cancelar

Tabla 36. Historia de usuario # 19

Historia de usuario	
Número: 19	Nombre: Modificar problema investigativo
Usuario: Responsable	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 16
Programador responsable: Abel Machado Monteso	
Descripción: El sistema permite al usuario poder acceder a la opción de modificar el problema a dar solución, donde se muestra una página con un formulario, con los campos siguientes que puede modificar: • Nombre • Área a la que tributa el problema • Línea de investigación a la que tributa el problema	
Observaciones: usuario autenticado	
Interfaz: Modificar problema investigativo Nombre: Aguas de la habana Área: Externa Línea de investigación: - Problema: ¿Cómo contribuir a la gestión de la información de la empresa Aguas de la Habana? Aceptar Cancelar	

Tabla 37. Historia de usuario # 20

Historia de usuario	
Número: 20	Nombre: Mostrar detalles del problema investigativo
Usuario: administrador, estudiante, profesor	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 17
Programador responsable: Abel Machado Monteso	
Descripción: El sistema permite al usuario poder mostrar los detalles de un problema seleccionado. El sistema mostrará una página con los datos del problema. • Nombre • Área a la que tributa el problema • Línea de investigación a la que tributa el problema	
Observaciones: usuario autenticado	
Interfaz: Detalles del problema investigativo Nombre: Aguas de la habana Área: Externa Línea de investigación: - Problema: ¿Cómo contribuir a la gestión de la información de la empresa Aguas de la Habana? Aceptar	

Tabla 38. Historia de usuario # 21

Historia de usuario	
Número: 21	Nombre: Adjuntar documento de tesis
Usuario: administrador, estudiante, profesor	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 43
Programador responsable: Abel Machado Monteso	
Descripción: El sistema debe permitirle a un usuario adjuntar un documento de tesis. El sistema debe permitir que los archivos a adjuntar sean en formato pdf o Word.	

Continúa en la próxima página

Tabla 38. Continuación de la página anterior

Observaciones: usuario autenticado
Interfaz: Adjuntar documento de tesis  Arrastrar y soltar o examinar Aceptar Cancelar

Tabla 39. Historia de usuario # 22

Historia de usuario	
Número: 22	Nombre: Descargar documento de tesis
Usuario: administrador, estudiante, profesor	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 44
Programador responsable: Abel Machado Monteso	
Descripción: El sistema debe permitir descargar un documento de tesis seleccionado por el usuario.	
Observaciones: usuario autenticado	
Interfaz:	

Tabla 40. Historia de usuario # 23

Historia de usuario	
Número: 23	Nombre: Añadir actas de los cortes de tesis
Usuario: administrador, profesor	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 20

Continúa en la próxima página

Tabla 40. Continuación de la página anterior

Programador responsable: Abel Machado Monteso
Descripción: El sistema debe permitirle al usuario adjuntar las actas de los cortes de tesis. Estas actas pueden estar en formato pdf o formato Word.
Observaciones: usuario autenticado
Interfaz:  Arrastrar y soltar o examinar

Tabla 41. Historia de usuario # 24

Historia de usuario	
Número: 24	Nombre: Crear acta de predefensa
Usuario: administrador, profesor	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 22
Programador responsable: Abel Machado Monteso	
Descripción: El sistema debe permitir crear el acta de la predefensa, para ello el usuario debe llenar los siguientes campos: • Tribunal • Fecha • Hora • Lugar donde sesiono el tribunal • Nombre de la tesis • Observaciones	
Observaciones: usuario autenticado	

Continúa en la próxima página

Tabla 41. Continuación de la página anterior

Interfaz:

Acta de Predefensa

Tribunal:

Fecha:

Hora:

Lugar :

Tesis:

Observaciones

Tabla 42. Historia de usuario # 25

Historia de usuario	
Número: 25	Nombre: Modificar acta de predefensa
Usuario: administrador, profesor	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 24
Programador responsable: Abel Machado Monteso	
Descripción: El sistema debe permitir modificar el acta de la predefensa, para ello el usuario debe seleccionar el acta que desea modificar y actualizar los campos: • Tribunal • Fecha • Hora • Lugar donde sesiono el tribunal • Nombre de la tesis • Observaciones	

Continúa en la próxima página

Tabla 42. Continuación de la página anterior

Observaciones: usuario autenticado	
Interfaz:	
Acta de Predefensa Tribunal: 1 Fecha: 28-11-2023 Hora: 0900 Lugar : Aula 201 Tesis: Sistema de gestión de trabajos de diploma de la facultad 4 Observaciones Debe corregir los aspectos que se mencionaron en la presentación y arreglar los aspectos del documento. Aceptar Cancelar	

Tabla 43. Historia de usuario # 26

Historia de usuario	
Número: 26	Nombre: Eliminar acta de predefensa
Usuario: administrador, profesor	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 23
Programador responsable: Abel Machado Monteso	
Descripción: El sistema debe permitir eliminar el acta de la predefensa, para ello el usuario debe seleccionar el acta que desea eliminar y el sistema muestra una advertencia “Desea eliminar el elemento seleccionado”	
Observaciones: usuario autenticado	

Continúa en la próxima página

Tabla 43. Continuación de la página anterior

Interfaz:

“Desea eliminar el elemento seleccionado”

Aceptar
Cancelar

Tabla 44. Historia de usuario # 27

Historia de usuario	
Número: 27	Nombre: Descargar acta de predefensa
Usuario: administrador, profesor	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 25
Programador responsable: Abel Machado Monteso	
Descripción: El sistema debe permitir descargar en pdf el acta de la predefensa, para ello el usuario debe seleccionar el acta que desea descargar	
Observaciones: usuario autenticado	
Interfaz: Acta de Predefensa Tribunal: 1 Fecha: 28-11-2023 Hora: 0900 Lugar : Aula 201 Tesis: Sistema de gestión de trabajos de diploma de la facultad 4 Observaciones Debe corregir los aspectos que se mencionaron en la presentación y arreglar los aspectos del documento. Descargar Aceptar Cancelar	

Tabla 45. Historia de usuario # 28

Historia de usuario	
Número: 28	Nombre: Mostrar detalles del acta de predefensa
Usuario: administrador, profesor	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 26
Programador responsable: Abel Machado Montes	
Descripción: El sistema debe permitir mostrar los detalles del acta, para ello el usuario debe seleccionar el acta que desea visualizar y se le muestra una ventana con los datos siguientes: • Tribunal • Fecha • Hora • Lugar donde sesiono el tribunal • Nombre de la tesis • Observaciones	
Observaciones: usuario autenticado	
Interfaz: Acta de Predefensa Tribunal: 1 Fecha: 28-11-2023 Hora: 0900 Lugar : Aula 201 Tesis: Sistema de gestión de trabajos de diploma de la facultad 4 Observaciones Debe corregir los aspectos que se mencionaron en la presentación y arreglar los aspectos del documento. Aceptar Cancelar	

Tabla 46. Historia de usuario # 29

Historia de usuario

Número: 29

Nombre: Buscar perfil de tesis

Usuario: administrador, profesor, estudiante

Prioridad en negocio: Prioridad

Riesgo en desarrollo: Alta

Puntos estimados: 1

Iteración asignada: 32

Programador responsable: Abel Machado Montes

Descripción: El sistema debe permitir buscar un perfil de tesis registrado en la base de datos

Observaciones: usuario autenticado

Interfaz:

Perfiles de tesis

Buscar

Autor	Nombre de la tesis	Tutores	Acciones
Abel Machado	Sistema para la gestión de las tesis en la Facultad 4	Mario Ramirez	

Aceptar

Cancelar

Tabla 47. Historia de usuario # 30

Historia de usuario	
Número: 30	Nombre: Buscar documento de tesis
Usuario: administrador, profesor, estudiante	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 34
Programador responsable: Abel Machado Montes	
Descripción: El sistema debe permitir buscar un documento de tesis registrado en la base de datos	
Observaciones: usuario autenticado	

Continúa en la próxima página

Tabla 47. Continuación de la página anterior

Interfaz:

Tesis

Buscar

Título	Autor	Tutores	Acciones
Sistema informático par..	Abel Machado	Juan Pérez	

Aceptar
Cancelar

Tabla 48. Historia de usuario # 31

Historia de usuario													
Número: 31	Nombre: Buscar usuario												
Usuario: administrador, profesor, estudiante													
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta												
Puntos estimados: 1	Iteración asignada: 35												
Programador responsable: Abel Machado Monteso													
Descripción: El sistema debe permitir buscar un usuario registrado en la base de datos													
Observaciones: usuario autenticado													
Interfaz: Usuarios Buscar <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: black; color: white;"> <th>Nombre y Apellidos</th> <th>Departamento</th> <th>usuario</th> </tr> </thead> <tbody> <tr> <td> </td> <td> </td> <td> </td> </tr> <tr> <td> </td> <td> </td> <td> </td> </tr> <tr> <td> </td> <td> </td> <td> </td> </tr> </tbody> </table> Aceptar Cancelar		Nombre y Apellidos	Departamento	usuario									
Nombre y Apellidos	Departamento	usuario											

Tabla 49. Historia de usuario # 32

Historia de usuario													
Número: 32	Nombre: Generar reporte de la evaluación del corte												
Usuario: administrador, profesor													
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta												
Puntos estimados: 1	Iteración asignada: 36												
Programador responsable: Abel Machado Monteso													
Descripción: El sistema debe permitir hacer un reporte a partir de la información administrada por las actas de los cortes. El sistema mostrará por tribunales las observaciones realizadas a las tesis, así como la evaluación de cada uno de los estudiantes													
Observaciones: usuario autenticado													
Interfaz: Evaluaciones Tribunal Elige <table><tr><th>Estudiante</th><th>Tesis</th><th>Observaciones</th></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr></table> Aceptar Cancelar		Estudiante	Tesis	Observaciones									
Estudiante	Tesis	Observaciones											

Tabla 50. Historia de usuario # 33

Historia de usuario	
Número: 33	Nombre: Generar reporte de perfiles existentes
Usuario: administrador, profesor	
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 37
Programador responsable: Abel Machado Montes	
Descripción: El sistema debe permitir hacer un reporte a partir de la información existente en la base de datos. El sistema mostrará una página con los perfiles existentes, de ellos se mostrará el nombre de la tesis, el autor, los tutores, y la opción de visualizar el documento	
Observaciones: usuario autenticado	

Continúa en la próxima página

Tabla 50. Continuación de la página anterior

Interfaz:

Perfiles de tesis

Autor	Nombre de la tesis	Tutores	Acciones
Abel Machado	Sistema para la gestión de las tesis en la Facultad 4	Mario Ramírez	
Pedro Junan Pérez	KbaCard	Mario Ramírez	
Juan Antonio Ramírez	Sistema de gestión para el suministro del agua en La Habana	Juan Pérez Pérez	

Aceptar
Cancelar

Tabla 51. Historia de usuario # 34

Historia de usuario													
Número: 34	Nombre: Generar reporte de estudiantes por Departamento												
Usuario: administrador, profesor													
Prioridad en negocio: Prioridad	Riesgo en desarrollo: Alta												
Puntos estimados: 1	Iteración asignada: 42												
Programador responsable: Abel Machado Montoso													
Descripción: El sistema debe permitir hacer un reporte a partir de la información existente en la base de datos. El sistema mostrará una página con los estudiantes por departamento, mostrando el nombre, la tesis, y el(los) tutores, así como el grupo al que pertenecen													
Observaciones: usuario autenticado													
Interfaz: Estudiantes por Departamento Grupo Elige Departamento Elige <table> <thead> <tr> <th>Nombre y apellido</th> <th>Departamento</th> <th>Tesis</th> </tr> </thead> <tbody> <tr><td> </td><td> </td><td> </td></tr> <tr><td> </td><td> </td><td> </td></tr> <tr><td> </td><td> </td><td> </td></tr> </tbody> </table> Aceptar Cancelar		Nombre y apellido	Departamento	Tesis									
Nombre y apellido	Departamento	Tesis											

Tabla 52. Historia de usuario # 35

Historia de usuario														
Número: 35		Nombre: Generar reporte de tutores por Departamento												
Usuario: administrador, profesor														
Prioridad en negocio: Prioridad		Riesgo en desarrollo: Alta												
Puntos estimados: 1		Iteración asignada: 41												
Programador responsable: Abel Machado Monteso														
Descripción: El sistema debe permitir hacer un reporte a partir de la información existente en la base de datos. El sistema mostrará una página con profesores que tienen el rol de tutor, por departamento, mostrando la cantidad de profesores por departamento, así como podrá visualizar la(s) tesis que tenga en su tutoría.														
Observaciones: usuario autenticado														
Interfaz: Tutores por Departamento Departamento de Informática: 2 Departamento de Inteligencia Computacional: 1 <table><tr><th>Nombre y apellido</th><th>Departamento</th><th>Tesis</th></tr><tr><td>Abel Machado</td><td>Informática</td><td>Ver</td></tr><tr><td>Abel Roque</td><td>Inteligencia Computacional</td><td>Ver</td></tr><tr><td>Pedro Juan Martínez</td><td>Informática</td><td>Ver</td></tr></table> Aceptar Cancelar			Nombre y apellido	Departamento	Tesis	Abel Machado	Informática	Ver	Abel Roque	Inteligencia Computacional	Ver	Pedro Juan Martínez	Informática	Ver
Nombre y apellido	Departamento	Tesis												
Abel Machado	Informática	Ver												
Abel Roque	Inteligencia Computacional	Ver												
Pedro Juan Martínez	Informática	Ver												

.4. Casos de pruebas de Aceptación

Tabla 53. Prueba de aceptación # 7

Caso de prueba de aceptación	
Código: HU47_P1	Historia de usuario: 47
Nombre: Eliminar tribunal	
Descripción: Eliminar tribunal de tesis	
Condiciones de ejecución: • El sistema debe estar en funcionamiento. • El usuario debe tener una cuenta registrada en el sistema con privilegios de administrador o de profesor. • El modelo debe de haber sido cargado y renderizado en el navegador.	

Continúa en la próxima página

Tabla 53. Continuación de la página anterior

Pasos de ejecución: El sistema muestra un listado de todos los tribunales de tesis creados y el usuario selecciona la que desea aceptar dando clic en el icono de eliminar
Resultados esperados: • El sistema debería permitir al usuario iniciar sesión con una cuenta de administrador. • El sistema debería permitir al usuario navegar hasta la sección del modelo. • El sistema debería permitir al usuario seleccionar el tribunal que desea eliminar. • Al hacer clic en “Eliminar” el sistema debería solicitar una confirmación. • Una vez confirmada la eliminación, el tribunal seleccionado debería eliminarse del modelo y notificar que se ha realizado la acción correctamente.

Tabla 54. Prueba de aceptación # 8

Caso de prueba de aceptación	
Código: HU6_P1	Historia de usuario: 6
Nombre: Eliminar rol	
Descripción: Eliminar rol	
Condiciones de ejecución: • El sistema debe estar en funcionamiento. • El usuario debe tener una cuenta registrada en el sistema con privilegios de administrador. • El modelo debe de haber sido cargado y renderizado en el navegador.	
Pasos de ejecución: El sistema muestra un listado de todos los roles creados y el usuario selecciona la que desea aceptar dando clic en el icono de eliminar	
Resultados esperados: • El sistema debería permitir al usuario iniciar sesión con una cuenta de administrador. • El sistema debería permitir al usuario navegar hasta la sección del modelo. • El sistema debería permitir al usuario seleccionar el rol que desea eliminar. • Al hacer clic en “Eliminar” el sistema debería solicitar una confirmación. • Una vez confirmada la eliminación, el tribunal seleccionado debería eliminarse del modelo y notificar que se ha realizado la acción correctamente.	

Tabla 55. Prueba de aceptación # 9

Caso de prueba de aceptación	
Código: HU13_P1	Historia de usuario: 13
Nombre: Eliminar perfil de tesis	
Descripción: Eliminar perfil de tesis	
Condiciones de ejecución: • El sistema debe estar en funcionamiento. • El usuario debe tener una cuenta registrada en el sistema con privilegios de administrador o de profesor. • El modelo debe de haber sido cargado y renderizado en el navegador.	
Pasos de ejecución: El sistema muestra un listado de todos los perfiles creados y el usuario selecciona el que desea eliminar aceptando dando clic en el icono de eliminar	
Resultados esperados: • El sistema debería permitir al usuario iniciar sesión con una cuenta de administrador o de profesor. • El sistema debería permitir al usuario navegar hasta la sección del modelo. • El sistema debería permitir al usuario seleccionar el perfil que desea eliminar. • Al hacer clic en “Eliminar” el sistema debería solicitar una confirmación. • Una vez confirmada la eliminación, el perfil seleccionado debería eliminarse del modelo y notificar que se ha realizado la acción correctamente.	

Tabla 56. Prueba de aceptación # 10

Caso de prueba de aceptación	
Código: HU15_P1	Historia de usuario: 15
Nombre: Eliminar problema investigativo	
Descripción: Eliminar problema investigativo	
Condiciones de ejecución: • El sistema debe estar en funcionamiento. • El usuario debe tener una cuenta registrada en el sistema con privilegios de administrador o de profesor. • El modelo debe de haber sido cargado y renderizado en el navegador.	
Pasos de ejecución: El sistema muestra un listado de todos los problemas investigativos creados y el usuario selecciona la que desea aceptar dando clic en el icono de eliminar	

Continúa en la próxima página

Tabla 56. Continuación de la página anterior

Resultados esperados: • El sistema debería permitir al usuario iniciar sesión con una cuenta de administrador o de profesor. • El sistema debería permitir al usuario navegar hasta la sección del modelo. • El sistema debería permitir al usuario seleccionar el problema que desea eliminar. • Al hacer clic en “Eliminar” el sistema debería solicitar una confirmación. • Una vez confirmada la eliminación, el tribunal seleccionado debería eliminarse del modelo y notificar que se ha realizado la acción correctamente.
--

Tabla 57. Prueba de aceptación # 11

Caso de prueba de aceptación	
Código: HU6_P1	Historia de usuario: 23
Nombre: Eliminar acta de predefensa	
Descripción: Eliminar acta de predefensa	
Condiciones de ejecución: • El sistema debe estar en funcionamiento. • El usuario debe tener una cuenta registrada en el sistema con privilegios de administrador. • El modelo debe de haber sido cargado y renderizado en el navegador.	
Pasos de ejecución: El sistema muestra un listado de todas las actas creadas y el usuario selecciona la que desea aceptar dando clic en el icono de eliminar	
Resultados esperados: • El sistema debería permitir al usuario iniciar sesión con una cuenta de administrador. • El sistema debería permitir al usuario navegar hasta la sección del modelo. • El sistema debería permitir al usuario seleccionar el acta que desea eliminar. • Al hacer clic en “Eliminar” el sistema debería solicitar una confirmación. • Una vez confirmada la eliminación, el tribunal seleccionado debería eliminarse del modelo y notificar que se ha realizado la acción correctamente.	

Tabla 58. Prueba de aceptación # 12

Caso de prueba de aceptación	
Código: HU28_P1	Historia de usuario: 28
Nombre: Eliminar acta de defensa	

Continúa en la próxima página

Tabla 58. Continuación de la página anterior

Descripción: Eliminar acta de defensa
Condiciones de ejecución: • El sistema debe estar en funcionamiento. • El usuario debe tener una cuenta registrada en el sistema con privilegios de administrador. • El modelo debe de haber sido cargado y renderizado en el navegador.
Pasos de ejecución: El sistema muestra un listado de todas las actas creadas y el usuario selecciona la que desea aceptar dando clic en el icono de eliminar
Resultados esperados: • El sistema debería permitir al usuario iniciar sesión con una cuenta de administrador. • El sistema debería permitir al usuario navegar hasta la sección del modelo. • El sistema debería permitir al usuario seleccionar el acta que desea eliminar. • Al hacer clic en “Eliminar” el sistema debería solicitar una confirmación. • Una vez confirmada la eliminación, el tribunal seleccionado debería eliminarse del modelo y notificar que se ha realizado la acción correctamente.

Tabla 59. Prueba de aceptación # 13

Caso de prueba de aceptación	
Código: HU5_P1	Historia de usuario: 5
Nombre: Crear rol	
Descripción: Crear rol	
Condiciones de ejecución: Usuario autenticado con permisos de administrador	
Pasos de ejecución: El usuario selecciona la opción de crear rol y rellena los campos requeridos.	
Resultados esperados: Se crea el nuevo rol y se muestra un listado de los roles creados en el sistema.	

Tabla 60. Prueba de aceptación # 14

Caso de prueba de aceptación	
Código: HU9_P1	Historia de usuario: 9
Nombre: Crear perfil de tesis	
Descripción: Crear perfil de tesis	

Continúa en la próxima página

Tabla 60. Continuación de la página anterior

Condiciones de ejecución: Usuario autenticado con permisos de administrador, profesor o estudiante
Pasos de ejecución: El usuario selecciona la opción de crear perfil de tesis y rellena los campos requeridos.
Resultados esperados: Se crea el nuevo perfil de tesis y se muestra un listado de los perfiles creados en el sistema.

Tabla 61. Prueba de aceptación # 15

Caso de prueba de aceptación	
Código: HU9_P1	Historia de usuario: 9
Nombre: Crear problema investigativo	
Descripción: Crear problema investigativo	
Condiciones de ejecución: Usuario autenticado con permisos de administrador, estudiante o profesor	
Pasos de ejecución: El usuario selecciona la opción de crear problema y rellena los campos requeridos.	
Resultados esperados: Se crea el nuevo problema investigativo y se muestra un listado de los problemas creados en el sistema.	

Tabla 62. Prueba de aceptación # 16

Caso de prueba de aceptación	
Código: HU22_P1	Historia de usuario: 22
Nombre: Crear acta de predefensa	
Descripción: Crear acta de predefensa	
Condiciones de ejecución: Usuario autenticado con permisos de administrador o profesor	
Pasos de ejecución: El usuario selecciona la opción de crear acta y rellena los campos requeridos.	
Resultados esperados: Se crea el acta y se muestra un listado con todas las existentes en el sistema.	

Tabla 63. Prueba de aceptación # 17

Caso de prueba de aceptación	
Código: HU27_P1	Historia de usuario: 27

Continúa en la próxima página

Tabla 63. Continuación de la página anterior

Nombre: Crear acta de defensa
Descripción: Crear acta de defensa
Condiciones de ejecución: Usuario autenticado con permisos de administrador o profesor
Pasos de ejecución: El usuario selecciona la opción de crear acta y rellena los campos requeridos.
Resultados esperados: Se crea el acta y se muestra un listado con todas las existentes en el sistema.

Tabla 64. Prueba de aceptación # 18

Caso de prueba de aceptación	
Código: HU3_P1	Historia de usuario: 3
Nombre: Modificar usuario	
Descripción: Modificar usuario	
Condiciones de ejecución: Usuario autenticado con permisos de administrador	
Pasos de ejecución: • El sistema debe estar en funcionamiento y accesible. • El usuario debe estar registrado y autenticado en el sistema. • El modelo debe de haber sido cargado y renderizado en el navegador.	
Resultados esperados: • El sistema debería permitir al usuario iniciar sesión con una cuenta de administrador. • El sistema debe dirigir al usuario a la sección de gestionar usuario. • El sistema debe permitir al usuario editar el formulario de datos. • Al hacer clic en “Aceptar”, el sistema debe cargar el usuario nuevo y confirmar que la modificación fue exitosa.	

Tabla 65. Prueba de aceptación # 19

Caso de prueba de aceptación	
Código: HU7_P1	Historia de usuario: 7
Nombre: Modificar rol	
Descripción: Modificar rol	
Condiciones de ejecución: Usuario autenticado con permisos de administrador	

Continúa en la próxima página

Tabla 65. Continuación de la página anterior

Pasos de ejecución: • El sistema debe estar en funcionamiento y accesible. • El usuario debe estar registrado y autenticado en el sistema. • El modelo debe de haber sido cargado y renderizado en el navegador.
Resultados esperados: • El sistema debería permitir al usuario iniciar sesión con una cuenta de administrador. • El sistema debe dirigir al usuario a la sección de gestionar rol. • El sistema debe permitir al usuario editar el formulario de datos. • Al hacer clic en “Aceptar”, el sistema debe cargar el rol nuevo y confirmar que la modificación fue exitosa.

Tabla 66. Prueba de aceptación # 20

Caso de prueba de aceptación	
Código: HU10_P1	Historia de usuario: 10
Nombre: Modificar perfil de tesis	
Descripción: Modificar perfil de tesis	
Condiciones de ejecución: Usuario autenticado con permisos de administrador o profesor	
Pasos de ejecución: • El sistema debe estar en funcionamiento y accesible. • El usuario debe estar registrado y autenticado en el sistema. • El modelo debe de haber sido cargado y renderizado en el navegador.	
Resultados esperados: • El sistema debería permitir al usuario iniciar sesión con una cuenta de administrador o profesor. • El sistema debe permitir al usuario editar el formulario de datos. • Al hacer clic en “Aceptar”, el sistema debe cargar el perfil de tesis y confirmar que la modificación fue exitosa.	

Tabla 67. Prueba de aceptación # 21

Caso de prueba de aceptación	
Código: HU48_P1	Historia de usuario: 48
Nombre: Modificar tribunal de tesis	
Descripción: Modificar tribunal de tesis	

Continúa en la próxima página

Tabla 67. Continuación de la página anterior

Condiciones de ejecución: Usuario autenticado con permisos de administrador o profesor
Pasos de ejecución: • El sistema debe estar en funcionamiento y accesible. • El usuario debe estar registrado y autenticado en el sistema. • El modelo debe de haber sido cargado y renderizado en el navegador.
Resultados esperados: • El sistema debería permitir al usuario iniciar sesión con una cuenta de administrador o profesor. • El sistema debe permitir al usuario editar el formulario de datos. • Al hacer clic en “Aceptar”, el sistema debe cargar el perfil de tesis y confirmar que la modificación fue exitosa.

Tabla 68. Prueba de aceptación # 22

Caso de prueba de aceptación	
Código: HU34_P1	Historia de usuario: 34
Nombre: Buscar documento de tesis	
Descripción: Buscar documento de tesis	
Condiciones de ejecución: Usuario autenticado	
Pasos de ejecución: • El sistema debe estar en funcionamiento y accesible. • El usuario debe estar registrado y autenticado en el sistema. • El modelo debe de haber sido cargado y renderizado en el navegador.	
Resultados esperados: • El sistema debería permitir al usuario buscar un documento de tesis. • El sistema debe permitir al usuario introducir el criterio de búsqueda y presionar en el botón de Buscar. • Al hacer clic en “Buscar”, el sistema debe cargar los resultados que coincidan con el criterio de búsqueda.	

Tabla 69. Prueba de aceptación # 23

Caso de prueba de aceptación

Continúa en la próxima página

Tabla 69. Continuación de la página anterior

Código: HU35_P1	Historia de usuario: 35
Nombre: Buscar usuario	
Descripción: Buscar usuario	
Condiciones de ejecución: Usuario autenticado	
Pasos de ejecución: • El sistema debe estar en funcionamiento y accesible. • El usuario debe estar registrado y autenticado en el sistema. • El modelo debe de haber sido cargado y renderizado en el navegador.	
Resultados esperados: • El sistema debería permitir al usuario buscar un usuario. • El sistema debe permitir al usuario introducir el criterio de búsqueda y presionar en el botón de Buscar. • Al hacer clic en “Buscar”, el sistema debe cargar los resultados que coincidan con el criterio de búsqueda.	

.5. Casos de prueba para las pruebas unitarias

Tabla 70. Caso de Prueba para el camino básico 1 (Fuente: Elaboración propia).

Proceso:
Crear tribunal.
Casos de prueba:
Crear tribunal. Escenario 1.1.
Camino independiente:
1,2,3,4,5
Entradas:
• Identificador: T1 • Presidente: Dr. Ignacio Pérez Reyes • Miembro: M.Sc. Omar Vázquez Palacio • Miembro: M.Sc. Omar Vázquez Palacio • Miembro: Ing. Marcos Remedio Tellez8
Resultados esperados:
Continúa en la siguiente página

-
- | |
|---|
| <ul style="list-style-type: none">• Adicionar un nuevo tribunal.• Mostrar el mensaje “Tribunal registrado satisfactoriamente”.• Muestra el listado de los tribunales creados. |
| Condiciones de ejecución: |
| <ul style="list-style-type: none">• Usuario autenticado y con los roles y permisos establecidos. |