Universidad de las Ciencias Informáticas

Facultad 2

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

Sistema informático para la gestión de la bibliografía física docente en la Facultad 2 de la Universidad de las Ciencias Informáticas

Autor: Yadian Tresgallos González

Tutores: MsC. Madelis Pérez Gil

Ing. Arianna Pérez Carmenates

La Habana, diciembre de 2023

Año 65 de la Revolución

“Quien piensa en fracasar, ya fracasó antes de intentar; quien piensa en ganar, lleva ya un paso adelante”.

Sigmund Freud

Dedicatoria

A las estrellas de mi universo, mis padres, por confiar siempre en mí, motivarme a convertirme en una mejor persona, por apoyar cada uno de mis pasos, por su entrega y dedicación y por ser sencillamente especiales, y a mi hermana, por convertirse en mi consejera y por estar siempre presente apoyando cada decisión en mi vida.

Agradecimientos

A la Universidad de las Ciencias Informáticas UCI, por ser el eslabón de todos en esta cadena ininterrumpida de aprehensiones.

A mis tutoras por alentarme a seguir superándome profesionalmente.

A mi familia, por ser parte de este proceso y apoyarme incondicionalmente.

A mis colegas y amigos de la UCI por su ayuda y orientación, sin la que hubiera sido imposible la culminación de esta investigación.

A los compañeros del grupo 2502 y del apartamento 136 105 que hemos sido uno en todo momento.

Declaración de Autoría

Declaramos ser autores de la presente tesis y reconocemos 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 _____.

_____

Yadian Tresgallos González

Autor

_____ _____

MsC. Madelis Pérez Gil

Tutora

Ing. Arianna Pérez Carmenares

Co-tutora

Resumen

En la actualidad, los sistemas de gestión son herramientas utilizadas para mejorar los procesos de negocio. Permiten a las organizaciones alcanzar sus metas al gestionar políticas y procedimientos de manera eficiente. La presente investigación se enfocó en analizar y comprender el proceso de gestión de entrega y devolución de libros en la Facultad 2 de la Universidad de las Ciencias Informáticas (UCI). Estos procesos se realizaban de forma manual, lo que resultaba en una ejecución lenta debido a la gran cantidad de información generada y almacenada. Dando lugar a la duplicidad de información, falta de veracidad y demoras en la atención a estudiantes y trabajadores. Por lo tanto, fue necesario implementar una solución que automatizara estos procesos. A partir de la premisa anterior se desarrolló una aplicación web utilizando la metodología XP para gestionar los libros presentes en el almacén de la Facultad 2. Se abordó en esta investigación sobre las herramientas utilizadas, el sistema de gestión de contenidos en el que se implementó el sistema y los elementos que se necesitaron para su construcción. El sistema incluye un catálogo de libros, información estadística sobre los libros en préstamos. La solución fue validada a través de pruebas, asegurando su correcto funcionamiento y la satisfacción del cliente.

Palabras clave: automatizara, libros, satisfacción del cliente, sistema de gestión.

Abstract

Currently, management systems are tools used to improve business processes. They allow organizations to achieve their goals by managing policies and procedures efficiently. This research focused on analyzing and understanding the management process of book delivery and return at Faculty 2 of the University of Informatics Sciences (UCI). These processes were carried out manually, resulting in slow execution due to the large amount of information generated and stored. This led to information duplication, lack of truthfulness, and delays in attention to students and workers. Therefore, it was necessary to implement a solution that automated these processes. Based on the premise above, a web application was developed using the XP methodology to manage the books present in Faculty 2's warehouse. This research covered the tools used, the content management system in which the system was implemented, and the elements needed for its construction. The system includes a book catalog, statistical information about the books on loan. The solution was validated through tests, ensuring its correct operation and customer satisfaction.

Keywords: *automated, books, customer satisfaction, management system.*

Índice general

Índice de figuras

Índice de tablas

Introducción

La bibliografía es una lista de fuentes de información que se utilizan en la investigación, escritura o elaboración de un trabajo académico, científico, literario o cualquier otro tipo de documento. Puede incluir libros, artículos de revistas, tesis, informes técnicos, sitios web, entrevistas y cualquier otra fuente de información que se haya utilizado en el trabajo. Por su parte, la bibliografía física se refiere a la lista de obras que se encuentran en un lugar físico específico, como una colección privada, una biblioteca o simplemente el material docente de estudiantes y profesores en un almacén de libros.

El desarrollo actual de las Tecnologías de la Información y las Comunicaciones (TIC) ha tenido fuerte impacto en el perfeccionamiento de diversos sistemas o herramientas informáticas que permiten gestionar los extensos volúmenes que se generan a diario de este tipo de bibliografía, empleada históricamente para el estudio, la investigación o preparación sobre un tema específico. Cuba no está exenta de las necesidades del mundo actual, por consiguiente, tampoco de la nueva revolución de la tecnología digital y de su utilidad en este ámbito.

La dirección del país ha trazado metas y estrategias para llevar a cabo el desarrollo de las TIC. La Universidad de las Ciencias Informáticas (UCI) es un proyecto creado por la Revolución Cubana, cuyo objetivo esencial es formar profesionales altamente calificados en la rama de la informática. La situación económica mundial y el costo que representa para el Ministerio de Educación Superior (MES) asumir la escasez de materiales de estudio de los estudiantes universitarios afecta, a nivel de país, la reimpresión de textos básicos en estado físico.

Específicamente en la Facultad 2 de la UCI, se estudian las carreras de Ingeniería en Ciencias Informáticas, Ingeniería en Ciberseguridad y el Programa de ciclo corto en Administración de Redes y Seguridad Informática. Actualmente no se cuenta con un sistema informático que controle la cantidad de libros extraídos del almacén por los estudiantes y trabajadores.

La entrega y recepción de libros que se lleva a cabo cada semestre; encontrándose a disposición de los estudiantes y profesores, tanto del curso regular diurno como el curso por encuentros, se va ejecutando durante el transcurso de las actividades docentes, ya que se puede solicitar un préstamo o devolver un libro en cualquier momento del curso.

Para mantener este control es necesario realizar un inventario donde se relacione la información referente a la cantidad de libros y títulos disponibles, así como la cantidad de libros en préstamo.

El proceso de actualización y control de la cantidad de libros en préstamos, así como la disponibilidad de libros en el local se realiza de forma manual y gradual a medida que se solicita un préstamo o se devuelve un libro, esto genera una considerable cantidad de documentos en formato duro para archivar debido a la cantidad de estudiantes y trabajadores que se atienden, lo cual conlleva a que en ocasiones se cometan errores al registrar o controlar un préstamo o devolución de un libro, hace que la labor de control, sea lenta, extensa y agotadora. Además, da origen a la duplicidad de la información y la pérdida de datos de la base material (libros).

Dada la situación anterior se plantea el siguiente **problema de investigación:** Las limitaciones que presenta el proceso de entrega y devolución de libros en la Facultad 2, influye negativamente en el control de los textos entregados y puede comprometer la integridad de los datos que se obtienen cuando se realiza un inventario. Se define como **objeto de estudio:** El proceso de gestión de bibliografía física docente.

Para dar solución al problema de investigación planteado, se define como **objetivo general:** Desarrollar un sistema informático que minorice las limitaciones que presenta el proceso de entrega y devolución de libros y asegure la integridad de la información generada durante el proceso, contribuyendo así a la gestión de la bibliografía física docente en la Facultad 2 de la Universidad de las Ciencias Informáticas.

Dentro del objeto de estudio de la investigación se precisa como **campo de acción:** El proceso de entrega y devolución de bibliografía física docente en la facultad 2 de la Universidad de las Ciencias Informáticas.

Con el propósito de dar cumplimiento a los objetivos establecidos, se elaboraron las siguientes **tareas de investigación**:

- Sistematización de los referentes teóricos que sustentan la investigación, relacionados con el uso de las herramientas informáticas para el proceso de gestión de entrega y recepción de libros en la Facultad 2 de la Universidad de las Ciencias Informáticas.
- Selección de las herramientas y tecnologías a utilizar para el desarrollo del sistema web que facilite el proceso de gestión de entrega y recepción de libros en la Facultad 2 de la Universidad de las Ciencias Informáticas.

- Especificación de los recursos funcionales para un sistema web que facilite el proceso de gestión de entrega y recepción de libros en la Facultad 2 de la Universidad de las Ciencias Informáticas.
- Implementación de un sistema web que garantice que se minoricen las limitaciones que presenta el proceso de entrega y recepción de libros en la Facultad 2 de la Universidad de las Ciencias Informáticas.
- Validación, a través de una estrategia de prueba, del sistema web que facilite el proceso de gestión de entrega y recepción de libros en la Facultad 2 de la Universidad de las Ciencias Informáticas.

Para guiar el cumplimiento del objetivo general planteado, se formulan las siguientes **preguntas científicas**:

- ¿Cuáles son los referentes teóricos relacionados con el uso de herramientas informáticas empleadas en el desarrollo del sistema para el proceso de gestión de entrega y recepción de libros en la Facultad 2 de la Universidad de las Ciencias Informáticas?
- ¿Cuál es el estado actual de las herramientas informáticas estudiadas en el desarrollo del sistema para el proceso de gestión de entrega y recepción de libros en la Facultad 2 de la Universidad de las Ciencias Informáticas?
- ¿Qué elementos deben tenerse en cuenta para llevar a cabo el análisis y diseño del sistema para el proceso de gestión de entrega y recepción de libros en la Facultad 2 de la Universidad de las Ciencias Informáticas?
- ¿Cómo materializar, en términos de componentes y código fuente, los elementos especificados en el sistema para el proceso de gestión de entrega y recepción de libros en la Facultad 2 de la Universidad de las Ciencias Informáticas?
- ¿Qué resultados se obtendrán al validar, a través de una estrategia de pruebas de software, el sistema para el proceso de gestión de entrega y recepción de libros en la Facultad 2 de la Universidad de las Ciencias Informáticas?

En el desarrollo de la presente investigación se emplearán los siguientes **métodos científicos**:

Métodos teóricos:

Analítico - Sintético: para el análisis de los elementos bibliográficos, definiciones y enfoques de diferentes autores en la sistematización del conocimiento sobre los sistemas de gestión, lenguajes, herramientas y metodologías para el proceso de control de la bibliografía física docente que posee el almacén de libros de la Facultad 2.

Modelación: para la realización de diagramas que evidencien la estructura y relaciones en el diseño de las clases necesarias para la implementación de la aplicación.

Métodos empíricos:

Entrevista: utilizada para recoger la información tanto primaria como secundaria que tributa a la estructura del estado actual del sistema de gestión de la bibliografía física docente en la Facultad 2 de la UCI.

Observación: permite conocer de manera práctica cómo se estructura y se gestionan los procesos de entrega y devolución de libros en las tres carreras de la Facultad 2, a partir de los resultados obtenidos en la práctica cotidiana.

El presente trabajo de diploma se ha estructurado en introducción, tres capítulos, conclusiones, recomendaciones, bibliografía y anexos. A continuación, un resumen de las diferentes temáticas que se abordan en los capítulos:

Capítulo 1: "Fundamentación teórica".

En este capítulo se expone la fundamentación teórica del trabajo, incluyéndose en el mismo, el estudio y análisis de sistemas homólogos existentes en la actualidad. Además, se define la metodología de software a seguir durante el proceso de desarrollo de software, los lenguajes, las herramientas y tecnologías a utilizar para la implementación de la propuesta de solución.

Capítulo 2: "Características del sistema y propuesta de solución".

Se realiza una descripción de la propuesta solución y teniendo en cuenta la metodología de software seleccionada, se realiza el análisis y diseño del sistema, además, de presentar los artefactos ingenieriles que documentan el proceso de desarrollo de software.

Capítulo 3: "Validación de la propuesta de solución".

En este capítulo se define la estrategia de pruebas a realizar para verificar el cumplimento de los requisitos del software. Además, se presentan los resultados de los casos de prueba, obteniendo una valoración integral de la solución propuesta que permita determinar los puntos débiles del sistema.

Capítulo 1

Fundamentación teórica

1.1. Introducción

El análisis de información se posiciona cada vez más como un factor clave para la toma de decisiones en una empresa, proporciona una base sólida de datos objetivos que permite la toma de decisiones basadas en la evidencia. El análisis de información implica la recopilación, procesamiento y análisis de datos relevantes sobre el negocio, el mercado, los clientes y la competencia.

En aras de comprender los elementos anteriormente planteados es necesario definir y analizar ciertos aspectos relevantes, que dan soporte teórico-metodológico y conceptual a la presente investigación.

Los siguientes epígrafes abarcan los conceptos de gestión de información y sistemas de gestión de información, fundamentales en la comprensión de la naturaleza y el alcance del proyecto, así como los sistemas de gestión de información y el proceso de entrega y devolución de bibliografía física en la UCI, junto a la realización de un estudio del estado del arte en cuanto a los sistemas informáticos de gestión de bibliografía física, compartiendo además los principales hallazgos y tendencias actuales en el ámbito.

Se presenta la metodología utilizada para el desarrollo del sistema informático, emprender demasiado desarrollo sin una metodología adecuada puede llevar a grandes pérdidas de tiempo y se detallan las herramientas y lenguajes informáticos utilizados en la implementación, las mismas permiten la creación de una aplicación web dinámica y eficiente que se ajuste a las necesidades de los usuarios.

1.2. Conceptos asociados al dominio del problema

Según la Real Academia Española, un libro es una obra científica, literaria o de cualquier otra índole con extensión suficiente para formar volumen, que puede aparecer impresa o en otro soporte (RAE, 2023).

Por su parte (Pérez, 2023) plantea que un libro es el conjunto de hojas de papel, vitela, u otra sustancia, manuscritas o impresas, colocadas en el orden en que se han de leer, y que reunidas o encuadernadas forman un volumen.

Al analizar estos conceptos se puede concluir que un libro es una obra que puede tener carácter científico, literaria o de cualquier otra índole, que se expresa a través de volúmenes de información que pueden tener un carácter impreso o digital.

Los libros en el proceso de enseñanza y aprendizaje constituyen un referente de imprescindible prioridad para los estudiantes, permitiendo la correcta formación de representaciones históricas, el desarrollo de la independencia cognoscitiva y la formación de métodos de trabajo científico. Por lo que constituyen un soporte bibliográfico para la impartición de las asignaturas.

En tal sentido el Ministerio de Educación Superior (MES) de la República de Cuba en su Resolución No.47/22 Reglamento Organizativo del Proceso Docente y de Dirección del Trabajo Docente y Metodológico para las Carreras Universitarias, establece en los artículos 140.1 y 147.1 las responsabilidades que tienen los colectivos de disciplinas y de asignaturas de proponer las bibliografías básicas y complementarias para la impartición de las asignaturas. Además, en los artículos 236 y 245 se establece que para el diseño de los programas de la disciplina y analíticos de las asignaturas la bibliografía forma parte de su estructura (MES, 2022).

Resulta necesario que las universidades lleven un control de los libros de textos que tienen en préstamo. Entiéndase por préstamo de un libro la autorización que se otorga a los usuarios para llevar fuera de la biblioteca, de manera gratuita, los libros de su interés a fin de que puedan leerlos en el momento y lugar que deseen (Martínez de Sousa, 1990).

Con el análisis realizado a estos conceptos se puede concluir que el préstamo de un libro es el movimiento de un libro de la biblioteca o local de almacenamiento, de manera autorizada, durante un periodo de tiempo determinado.

1.3. Gestión de información

El tránsito de la humanidad a la sociedad de la información es un proceso histórico que ha transformado significativamente la forma en que las personas interactúan, trabajan y se comunican entre sí. Ha sido impulsado por varios factores, incluyendo el desarrollo y la adopción de nuevas tecnologías de la información y la comunicación, como Internet, la computación en nube y las redes sociales. Estas tecnologías permiten el acceso a información instantánea, la conexión con personas de todo el mundo y la realización de transacciones comerciales de manera virtual.

Esto ha provocado que muchas instituciones, en el aspecto informativo, presenten una excesiva centralización de la información y un flujo abundante de documentos impresos y digitales. Además, quienes necesitan la información no disponen de ella en el momento y espacio adecuados. Con frecuencia se observa que directivos y trabajadores se encuentran abrumados por documentos e informaciones innecesarias.

La Gestión de Información (GI) surge como concepto dentro del campo de la Ciencia de la Información, orientado al manejo de la Inteligencia Corporativa de una organización. Es el proceso de planificar, coordinar, controlar y evaluar el uso, la distribución y el almacenamiento de información dentro de una organización o sistema. Está destinada a mejorar la eficiencia, la eficacia, la seguridad y el valor de la información para los usuarios (Rodríguez Salas, 2002).

Se enfoca en todas las formas de información, incluyendo la información en papel, digital, electrónica y verbal. Un enfoque clave es la creación de políticas y estrategias para garantizar que la información se almacene y se distribuya de manera segura y en el formato correcto. Los sistemas de información juegan un papel importante en la gestión de la información, ya que permiten la recopilación, el almacenamiento y la distribución de datos e información de manera eficiente. También pueden ser utilizados para automatizar procesos de negocio y mejorar la eficiencia y eficacia de las operaciones (Vidal Ledo, 2012).

La gestión de la información resulta un proceso crítico para garantizar que la información sea almacenada, organizada y distribuida de manera eficiente y segura. Los sistemas de información son herramientas clave en la gestión de la información, y las medidas de seguridad deben ser implementadas para proteger la información de amenazas internas y externas.

1.4. Sistemas para la gestión de información

Antes de definir un Sistema para la Gestión de Información (SGI) es necesario conocer primero algunos aspectos que caracterizan a los Sistemas de Información (SI).

Los sistemas de información pueden estar compuestos por personas, hardware, software, datos, procedimientos y redes de comunicación que permiten la captura, procesamiento, almacenamiento y distribución de información dentro de una organización. Estos sistemas pueden ser utilizados por diferentes partes de la organización para manejar diferentes tipos de información, incluyendo información financiera, de recursos humanos, operativa y estratégica (Tramullas, 1997).

Pueden ser utilizados en cualquier tipo de organización, desde pequeñas empresas hasta grandes corporaciones y organismos gubernamentales. Algunos ejemplos de sistemas de información comunes incluyen sistemas de gestión de bases de datos, sistemas de gestión de contenidos, sistemas CRM (Customer Relationship Management o en español, Gestión de Relaciones con el Cliente), sistemas ERP (Enterprise Resource Planning o en español, Planificación de Recursos Empresariales) y sistemas de información geográfica.

Dentro de este preámbulo se encuentran entonces los SGI, ya que el término, automatización/informatización de los procesos claves de una organización, generalmente viene aparejado a la implantación de un SGI, siendo este el objetivo fundamental que persiguen los mismos (Bootello, 2007-2008).

Teniendo en cuenta los elementos anteriormente planteados, la definición de SGI para la presente investigación se ajusta al concepto dado por la Organización Internacional de Normalización (ISO). El cual comprende que un SGI es un "sistema compuesto de equipos y de personal pertinente que realiza funciones de entrada, procesamiento, almacenamiento, salida y control de información, con el fin de llevar a cabo una secuencia de operaciones con datos" (Bootello, 2007-2008).

Se pueden clasificar en diferentes tipos, como sistemas transaccionales, sistemas de apoyo a la toma de decisiones, sistemas de inteligencia empresarial, sistemas de gestión de contenido, sistemas de gestión de bases de datos y sistemas de gestión de documentos. Cada tipo de sistema tiene funcionalidades específicas que permiten a las organizaciones la gestión de diferentes tipos de información, estos sistemas pueden ser personalizados para satisfacer las necesidades específicas de una empresa y son utilizados en diversos sectores como finanzas, ventas, marketing, recursos humanos y logística (Bootello, 2007-2008).

1.5. SGI y proceso de entrega y devolución de bibliografía física en la UCI

Analizados los elementos anteriores, no queda duda que el uso de los SGI resulta recomendable en organizaciones que se orienten al tema principal de este trabajo, el proceso de entrega y devolución de bibliografía física. No existe mucha información referida a la relación de estos dos elementos, más allá de que uno se beneficia del otro. Pero se hace realmente necesario un análisis de esta relación, ya que son los componentes esenciales del presente trabajo de diploma.

Las acciones de entrega y devolución de libros generan datos que no siempre son difíciles de almacenar y manejar por parte de los trabajadores. Pero esto solo ocurre

cuando se está en presencia de un negocio pequeño. A medida que el negocio crece y se multiplican los clientes y los factores que intervienen en la prestación del servicio, el volumen de la información generada aumenta considerablemente. Por lo que se hace más complicado, el realizar los registros de extracciones y devoluciones de textos.

Orientado a resolver este tipo de problemas, una buena solución es la puesta en práctica de un SGI. Basta con incorporarlo al flujo de trabajo, para abstraerse del problema que significa almacenar y administrar la información, y dar prioridad a realizar acciones que maximicen el rendimiento a partir de los resultados que el SGI genere.

Las personas que intervienen en estos procesos son: la vicedecana de formación, el asistente de la vicedecana (supervisor o responsable de almacén) y los estudiantes y trabajadores de la facultad. La descripción de los roles se encuentra en el Anexo 1.

El actuar diario de la entrega y devolución de bibliografía está basado en la recepción de solicitudes, para su posterior realización. Por tanto, la información referente a esta etapa del proceso debe quedar resguardada de manera que facilite su consulta y garantice su seguridad; elementos que, por las vías tradicionales, como papeles o documentos digitales, resultan de difícil acceso. No siendo así con un SGI, ya que, al poseer la información incluida dentro de su propio funcionamiento, y contar con herramientas y tecnologías informáticas que cooperan entre sí, bajo principios de interoperabilidad, posibilitan la accesibilidad y la seguridad de la información.

1.6. Estudio del estado del arte

Luego de una revisión exhaustiva y crítica de la literatura existente sobre sistemas informáticos que constituyen los representantes más notorios de esta categoría de aplicación, a continuación, se analizan los mismos, haciendo énfasis en determinados aspectos que puedan ser de utilidad para esta investigación.

1.6.1. SGI en el proceso de entrega y devolución de bibliografía física

Actualmente existen varios sistemas informáticos para la gestión de bibliografías y préstamos de libros. A continuación, se mencionan algunos de los más relevantes:

CodeAchi Library Management System

Library Management System (Sistema de Gestión de Bibliotecas) fue desarrollado por CodeAchi (compañía de software en la India) en 2019 y se mantiene en constante actualización. Es un software para la gestión digital de bibliotecas, satisface cada consulta individual relacionada con la información de su biblioteca. Es un sistema de

pago, aunque presenta cierta flexibilidad a la hora de comprarlo. Presenta prioridad de soporte, posventa y seguridad de los datos. Entre sus funciones principales se encuentran: búsqueda, categorización, gestión de archivos y gestión de usuarios. Este software solo es compatible con los sistemas operativos, Windows 7 o superiores (Capterra, CodeAchi Library Management System, 2019).

Resource Mate

Este software centra sus energías en satisfacer las necesidades de las bibliotecas pequeñas y medianas y los desafíos de la automatización bibliotecaria. La empresa se inició en 1991 en Canadá y se constituyó en 1993 con el lanzamiento oficial de ResourceMate®. En 2017, Harris Computer Systems (comprador de soluciones de software, cuyo objetivo es adquirir empresas con potencial de crecimiento) adquirió ResourceMate® y continúa creciendo y desarrollándose en la actualidad (ResourceMate, 2023).

Es un software de gestión bibliotecaria diseñado para bibliotecas, escuelas, lugares de culto, comunidades de jubilados, establecimientos penitenciarios, museos, gobiernos, centros médicos o de enfermería, corporaciones y organizaciones sin ánimo de lucro. Entre sus funciones principales se encuentran: el préstamo y la devolución, búsqueda, categorización, gestión de archivos, contenidos, usuarios y circulación y agrupación. Además, permite administrar libros de textos y mostrar información sobre la actualización de los cambios realizados. Es un software de carácter propietario, con facilidad de pago. Solo puede ser utilizado en países como Canadá, Estados Unidos, Irlanda y Reino Unido (Capterra, ResourceMate, 2020).

Estos sistemas informáticos tienen en común la gestión de bibliografías, pero podrían variar en cuanto a su capacidad de integración con otras herramientas y servicios, la personalización de la interfaz y la facilidad de uso.

En resumen, existen numerosos sistemas informáticos para la gestión de bibliografías, todo esto puede aportar valiosa información para la elaboración de la tesis sobre un sistema informático para la gestión de entrega y devolución de libros en la Facultad 2 de la Universidad de Ciencias Informáticas de Cuba.

1.6.2. SGI en el proceso de entrega y devolución de bibliografía física en Cuba

Biblioteca virtual

Es una herramienta web creada en la Universidad “Carlos Rafael Rodríguez” de Cienfuegos en 2010. Automatiza los procesos sustantivos que se originan en la

biblioteca en cuestión; dígase selección y adquisición de libros, préstamos, catalogación y clasificación de las fuentes bibliográficas que se reciben en la biblioteca y estadísticas que se generan (Cañedo Iglesias, Zamora Fonseca, Linares Armas, Martínez Santos, & Nuñez Chaviano, 2010).

El sistema informático organiza el trabajo atendiendo a las diferentes áreas que cuenta la biblioteca, facilitando así un mayor control para acceder a la información, permite un preciso control estadístico, cuenta con facilidades de búsquedas y una interfaz fácil de utilizar por cualquier usuario. Es importante destacar que gracias a este sistema se definen correctamente algunas áreas que no existían como es el caso del procesamiento estadístico (Cañedo Iglesias, Zamora Fonseca, Linares Armas, Martínez Santos, & Nuñez Chaviano, 2010).

A pesar de contar con grandes ventajas, sobre todo en el área estadística de una biblioteca, este sistema no cumple con todos los requisitos deseados pues no tiene la posibilidad de realizar reportes, requisito fundamental de la nueva aplicación para el correcto control de los procesos presentes en el almacén de libros de la facultad.

Herramienta web para la gestión de la base material de estudio de la Universidad de Pinar del Rio “Hermanos Saíz Montes de Oca”

Herramienta web diseñada en 2013 que abarca los procesos de actualización general de los inventarios de la base material de estudio (textos), gestión de las entradas y salidas de la base material de estudio y la gestión de préstamos y devoluciones de la base material de estudio a estudiantes y profesores (Curbelo Carmona, 2013).

Proporciona un inventario general actualizado de la base material de estudio y su estado real en todas las áreas de la universidad, además da a conocer la existencia y ubicación de la base material de estudio según el plan bibliográfico de cada carrera, años y semestre, lo que favorece gestionar los movimientos e intercambios necesarios de la misma (Curbelo Carmona, 2013).

A pesar de que la herramienta web para la gestión de la base material de estudio en la Universidad de Pinar del Río, cuenta con un módulo para gestionar el préstamo a estudiantes y profesores y un módulo para generar un inventario actualizado sobre la base material de estudio, no cuenta con un módulo para realizar el reporte del estado de extracción de los libros, además la herramienta no genera un modelo de entrega y devolución de libros, el cual es la ficha que utiliza el almacén de libros de la Facultad 2 para mantener un control sobre el proceso de entrega y devolución de textos para cada estudiante y profesor.

A continuación, se presenta un resumen donde se recogen los datos más significativos de cada uno de los sistemas analizados:

Tabla 1: Resumen de sistemas homólogos

Herramienta	Código abierto	Multiplataforma	Gestión de prestamos	Exportar información personalizada
CodeAchi Library Management System	No	No	Si	No
Resource Mate	No	No	Si	No
Biblioteca virtual	Si	Si	Si	No
Herramienta web para la gestión de la base material de estudio	Si	Si	Si	No

Los sistemas estudiados en el ámbito internacional tienen como desventaja que el acceso a su código fuente es privativo, por lo que no cumplen con el paradigma de independencia tecnológica que hoy aboga el país. Sistemas como: CodeAchi Library Management System y Resource Mate no son multiplataforma. En su totalidad los sistemas carecen de la funcionalidad de exportar información, siendo esto de gran importancia para el negocio.

1.6.3. Conclusiones del estudio del estado del arte

El estudio sobre sistemas informáticos de gestión de bibliografía física arrojó que es evidente la necesidad de implementar soluciones innovadoras y eficientes para mejorar la gestión de préstamos y devoluciones dentro de las bibliotecas universitarias.

El estado del arte muestra algunos ejemplos de sistemas informáticos de gestión de bibliografías disponibles a nivel internacional y nacional, y los desafíos y oportunidades existentes en la implementación de soluciones personalizadas. En consecuencia, el sistema informático propuesto para la Facultad 2 permitirá una mejor gestión de los procesos de préstamo y devolución de libros, de manera eficiente y adaptada a las necesidades específicas de la comunidad universitaria.

1.7. Metodología de desarrollo del software

Las metodologías de desarrollo de software son técnicas y enfoques utilizados en la gestión y desarrollo de proyectos de software con el objetivo de controlar el proceso de desarrollo y garantizar la calidad del producto final (PRESSMAN, 2001).

En la actualidad existen varias metodologías divididas en dos grandes grupos, las tradicionales y las ágiles. Las primeras se centran en elaborar una documentación exhaustiva de todo el desarrollo del software y en cumplir con la planificación realizada en la fase inicial del proyecto, sin dar margen a posibles cambios, puesto que al ocurrir uno, se ven afectados varios componentes del proceso de desarrollo del software. Otro elemento que las caracteriza es la gran cantidad de participantes con los que cuenta, pues requiere de un equipo de trabajo capaz de administrar un proceso complejo en varias etapas (Almarales Lara & Sencial Terrero, 2015).

Cada metodología tiene sus ventajas y desventajas, y la elección de la adecuada depende de las necesidades y requisitos específicos del proyecto. Lo importante es seleccionar la que mejor se adapte al equipo de desarrollo y al proyecto en sí mismo.

Con el fin de aprovechar los beneficios que proporcionan los procesos de construcción rápida de software, según las características del software solicitado, dada la premura de tiempo en que se debería construir y los requisitos de alto nivel, se tomó la decisión de utilizar un modelo de desarrollo de tipo ágil; dado que estos métodos permiten la implementación con el mínimo de documentación, en el menor tiempo posible y teniendo al cambio como una constante.

1.7.1. Programación Extrema

La metodología de Programación Extrema (XP) es una metodología ágil de desarrollo de software, que pone énfasis en la simplicidad, la comunicación y la retroalimentación constante entre los miembros del equipo.

Está centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje del equipo de desarrollo y propiciando un buen clima de trabajo. Se basa en la retroalimentación 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. Se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, donde existe un alto riesgo técnico (Beck, 2000).

XP aboga por disminuir la mítica curva exponencial del costo del cambio a lo largo del proyecto, lo suficiente para que el diseño evolutivo funcione. Esto se consigue gracias a las tecnologías disponibles para ayudar en el desarrollo de software y a la aplicación disciplinada de doce prácticas, de las cuales, las que hacen de XP la metodología idónea para esta tesis son las siguientes (Canós, Letelier, & Carmen Penadés, 2011):

- **Definición de requerimientos o historias de usuario:** Lo primero sería identificar las necesidades del sistema a través de reuniones con el supervisor del almacén, los estudiantes y otros usuarios potenciales del sistema, se puede realizar con técnicas de entrevistas. Estos requerimientos deben ser expresados como historias de usuario, las cuales deben ser lo suficientemente detalladas para comprender el problema que se está resolviendo.
- **Diseño arquitectónico:** Será necesario diseñar la arquitectura software, según las metodologías de diseño simples y elegantes, y que se ajusten a los requerimientos del software.
- **Pruebas de desarrollo:** Es necesario realizar pruebas continuas de las funcionalidades implementadas, tanto en unidades aisladas, como en el sistema final integrado. Esto asegura que el software continúe cumpliendo con los requerimientos iniciales.
- **Integridad continua:** Se integrarán las secciones terminadas del proyecto frecuentemente para evitar problemas y resolver conflictos durante el proceso.
- **Lanza versiones incrementales:** El sistema debe ser entregado por piezas o versiones parciales de forma incremental, lo que permite que el cliente tenga un prototipo del sistema y pueda evaluar si cumple con sus expectativas y necesidades.
- **Comunicación constante:** Los miembros del equipo deben mantener una comunicación frecuente y efectiva entre sí y con los profesores o tutores, a través de reuniones para revisar el progreso del proyecto.
- **Retroalimentación:** Debe fomentarse la retroalimentación constante del usuario, a través de evaluaciones y pruebas, para verificar el correcto funcionamiento del sistema y sugerir mejoras.

El ciclo de vida ideal de XP es uno de los elementos más polémicos y sujetos a debate de la metodología. Mientras que Kent Beck –su fundador– la concibe como un conjunto de prácticas agrupadas en seis faces específicas: exploración, planificación de entregas, iteraciones, producción, mantenimiento y muerte del proyecto. Otros autores como Manuel Calero Solís (Calero Solís, 2003), Gerardo Fernández Escribano (Fernández Escribano, 2002) y Doug Wallace, Isobel Raggett y Joel Aufgang (Wallace,

Raggett, & Aufgang, 2003) agrupan las prácticas definidas por Beck en cuatro etapas: planificación, diseño, implementación y pruebas.

Para la selección de las fases de la metodología en la presente investigación se tuvieron en cuenta dos elementos fundamentales: el objetivo de minimizar el tiempo requerido para la elaboración de los artefactos pertenecientes a cada una de las fases y que solo una persona llevará a cabo todas las tareas relacionadas con la elaboración del sistema. Se definen así específicamente cuatro fases: exploración, planificación, diseño e implementación y pruebas.

XP presenta determinados artefactos para guiar el desarrollo del proyecto. Estos son las Historias de Usuario (HU), las Tareas de Ingeniería (TI) y las tarjetas Clase-Responsabilidad-Colaboradores (CRC).

Las HU son la técnica utilizada para especificar los requisitos del software. Se trata de tarjetas en las cuales el cliente describe brevemente las características que el sistema debe poseer. El tratamiento de las HU es muy dinámico y flexible. Cada HU es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en poco tiempo (Beck, 2000).

Las TI constituyen la base de la realización concreta de las HU. Una HU puede generar una o más TI, dependiendo de la complejidad de la misma. En las TI deben estar presentes los datos necesarios para que el programador realice la implementación del código en el tiempo estimado (Beck, 2000).

Las tarjetas CRC constituyen un artefacto que define la estructura del proyecto y las relaciones entre las clases. Estas tarjetas se dividen en tres secciones que contienen la información del nombre de la clase, sus responsabilidades y sus colaboradores. Una clase es cualquier persona, evento, concepto, pantalla o reporte. Las responsabilidades de una clase son los elementos que conoce y las que realiza, sus atributos y métodos. Los colaboradores de una clase son las demás clases con las que trabaja en conjunto para llevar a cabo sus responsabilidades (Beck, 2000).

1.8. Herramientas y lenguajes informáticos

La selección de las herramientas y tecnologías en el desarrollo de un proyecto, es un punto sumamente importante. Definen la naturaleza de la solución y son la base de la calidad de la misma. A continuación, se describen las herramientas informáticas y lenguajes utilizados en la implementación.

1.8.1. Lenguajes

1.8.1.1. Lenguajes de modelado

Un lenguaje de modelado provee un vocabulario y conjunto de reglas centradas en la representación conceptual y física de un sistema. El Lenguaje Unificado de Modelado (Unified Modeling Language), es un lenguaje informático, gráfico o textual que sigue un conjunto semántico de reglas y marcos para el diseño y construcción de estructuras y modelos. Se utiliza para visualizar, especificar, construir y documentar software orientados a objetos. Permite a los creadores de sistemas generar diseños que capturen sus ideas en una forma fácil de comprender y comunicar a otras personas (Amaya, 2017).

La elección de **UML v2.5** se sustenta en su facilidad para especificar, documentar, visualizar y comprender los artefactos generados en las diferentes fases del desarrollo de software, así como sus ventajas para la comunicación entre el cliente y el equipo de desarrollo.

BPMN 2.0 (Business Process Model and Notation, o Notación para el Modelado de Procesos de Negocio) es una notación estándar de modelado de procesos de negocio, utilizado para describir y visualizar diferentes procesos de negocio y flujos de trabajo mediante una representación gráfica fácil de entender. La notación es utilizada por analistas de negocios, desarrolladores y otros profesionales para modelar, analizar y mejorar procesos de negocio (White, 2011).

El principal objetivo de BPMN es proporcionar una notación estándar que sea fácilmente legible y entendible por parte de todos los involucrados e interesados del negocio. Entre estos interesados están los analistas de negocio, los desarrolladores técnicos y los gerentes y administradores del negocio. El uso de BPMN tiene la finalidad de servir como lenguaje común para cerrar la brecha de comunicación que frecuentemente se presenta entre el diseño de los procesos de negocio y su implementación.

1.8.1.2. Lenguaje de marcado de hipertexto

HTML5 es la última versión de HTML (Hypertext Markup Language, o Lenguaje de Marcado de Hipertexto) utilizado para crear y estructurar el contenido de los sitios web. Se diseñó para ser más fácil de escribir, leer y mantener que las versiones anteriores de HTML. Es un estándar a cargo de la World Wide Web Consortium (W3C),

organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación (Hickson & Hyatt, 2009).

Se utiliza esta versión que ha agregado nuevas características y funcionalidades para facilitar su uso y mejoran la calidad visual y la interactividad de las páginas web, haciendo posible la creación de sitios web más estilizados, interactivos y adaptables a diferentes tipos de dispositivos.

1.8.1.3. Lenguaje de diseño gráfico

CSS3 (Cascading Style Sheets, u Hojas de Estilo en Cascada) es un lenguaje de estilo utilizado para personalizar el aspecto visual de páginas web escritas en HTML. Esta tecnología se utiliza para proporcionar estilos y diseños visuales a los sitios web, lo que ayuda a mejorar la apariencia y la experiencia del usuario (Barcia, 2003).

CSS funciona descifrando todos los elementos de una página web (como los encabezados, los párrafos, las imágenes y los botones) y aplicando estilos específicos a cada uno de ellos, por lo que mejora la apariencia visual y la legibilidad de la página.

Se emplea CSS3, como tercera versión importante de cara a los desarrolladores de webs, por la incorporación de nuevos mecanismos para mantener un mayor control sobre el estilo con el que se muestran los elementos de las páginas, sin tener que recurrir a trucos o hacks, que a menudo complicaban el código de las webs.

1.8.1.4. Lenguajes de programación

JavaScript es un lenguaje de programación de alto nivel utilizado principalmente para crear aplicaciones web interactivas y dinámicas. Se utiliza con frecuencia en conjunto con HTML y CSS para mejorar la experiencia del usuario en las páginas web (González & Gaudioso, 2001).

PHP v8.1.23 (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML.

Lo que distingue a PHP de algo del lado del cliente como Javascript es que el código es ejecutado en el servidor, generando HTML y enviándolo al cliente. El cliente recibirá el resultado de ejecutar el script, aunque no se sabrá el código subyacente que era. El servidor web puede ser configurado incluso para que procese todos los ficheros HTML con PHP. Se emplea PHP por su extrema simplicidad, pero a su vez ofrece muchas características avanzadas para los programadores (PHP, 2023).

Se emplea este lenguaje de programación porque permite la creación de páginas web interactivas y dinámicas, además puede agregar animaciones, validación de formularios, efectos visuales y aplicaciones web en tiempo real.

1.8.2. Herramientas y tecnologías

1.8.2.1. Herramienta para el modelado del sistema

Visual Paradigm v16.2 es una herramienta CASE (Computer Aided Software Engineering o en español, Ingeniería de Software Asistida por Computadora) que permite el modelado y diseño de sistemas de software a través de lenguaje de modelado UML.

Esta herramienta proporciona una amplia gama de características, que incluyen la creación de diagramas UML específicos, como diagramas de casos de uso, diagramas de clases, diagramas de objetos, diagramas de actividades y diagramas de secuencia. Además, permite la generación automática de código en varios lenguajes de programación, así como la integración con otras herramientas de desarrollo de software.

Visual Paradigm es gratuito, aunque se encuentra bajo una licencia que no permite su modificación o venta, ajustándose esta última a las políticas de soberanía tecnológica trazadas por el país. Teniendo en cuenta la importancia del uso del software libre en Cuba, la existencia de una Licencia UCI para el uso de Visual Paradigm y las características de esta herramienta, se selecciona, en su versión 16.2, para realizar el modelado del sistema.

Aunque ningún artefacto de la metodología seleccionada requiere el uso de esta herramienta, se utilizó para una mejor comprensión del sistema a través del modelado del negocio empleando la notación estándar de modelado de procesos de negocio BPMN 2.0.

1.8.2.2. Entorno de desarrollo integrado

Un entorno de desarrollo integrado o IDE (Integrated Development Environment) es un espacio de trabajo virtual que se utiliza para el desarrollo y programación de aplicaciones de software. Gracias a las herramientas y mecanismos que aporta un IDE la tarea de programar es mucho más sencilla, ahorrando tiempo y consiguiendo que la productividad y eficiencia de los programadores y desarrolladores sea mucho más alta (Felipe, 2021)

Se decide utilizar **Visual Studio Code v1.82.2** porque es un editor de código fuente desarrollado por Microsoft, muy popular entre los desarrolladores de software de todo el mundo. Incluye varias mejoras y características nuevas como soporte mejorado para varios lenguajes de programación, una amplia variedad de extensiones y herramientas de desarrollo integradas.

1.8.2.3. Software de virtualización

VirtualBox v7.0 es un software de virtualización de código abierto que permite crear y ejecutar máquinas virtuales en un ordenador. Desarrollado por Oracle, es compatible con varios sistemas operativos, como Windows, macOS, Linux y Solaris. En otras palabras, si se tiene un ordenador con Windows, GNU/Linux o incluso macOS, puede crear una máquina virtual con cualquier otro sistema operativo para utilizarlo dentro del nativo.

1.8.2.4. Sistema gestor de base de datos

Data Base Management System (DBMS) o Sistema de Gestión de Base de Datos (SGBD), es un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. Un SGBD permite definir los datos a distintos niveles de abstracción y manipular dichos datos, garantizando la seguridad e integridad de los mismos (Marín, 2019).

MariaDB v10.5.19 es una alternativa a MySQL de código abierto y gratuita que ofrece una mayor eficiencia, un mejor rendimiento de la base de datos y soporte para varios tipos de datos mediante múltiples motores de almacenamiento. Brinda un rendimiento más rápido y más motores de almacenamiento, lo que le permite almacenar diferentes tipos de datos.

Para el desarrollo del sistema se decide utilizar MariaDB debido a que los sistemas que lo utilizan se benefician de una mayor resistencia y tiempos de desarrollo más cortos. Es atómico y transaccional, por lo que puede evitar eventos de éxito parcial, si una operación en una transacción falla, las exitosas también se revierten, devolviendo la base de datos a su estado original (MariaDB, 2023).

1.8.2.5. Servidor web

Un servidor web es un programa informático que procesa aplicaciones realizando conexiones bidireccionales y/o unidireccionales, síncronas o asíncronas con el cliente, generando o cediendo una respuesta en cualquier lenguaje o aplicación del lado del cliente. Es quien gestiona la mayor parte de las funciones de lógica de negocio y de acceso a los datos de la aplicación. Los principales beneficios de la aplicación de la tecnología de servidores web son la centralización y la disminución de la complejidad en el desarrollo de aplicaciones (de Souza, 2019).

Apache v2.4.56 es un servidor web que implementa el protocolo HTTP, es software libre y se puede instalar en los sistemas operativos Windows y Linux. Es de fácil configuración, pues su estructuración en módulos permite al usuario utilizar los servicios y funcionalidades que ofrece.

Se define Apache como software para el servidor de aplicaciones, por su flexibilidad, rapidez, porque es gratuito, multiplataforma e interpreta varios lenguajes como Python y HTML. Estas características promueven la eficiencia y rendimiento del sistema que debe ser capaz de dar respuestas a las peticiones con un nivel aceptable de desempeño, teniendo en cuenta la concurrencia que pueda existir; debe prestar servicios sin que se amplíen los rangos de tiempo de repuesta (Apache, 2023).

1.8.2.6. Sistema de gestión de contenidos

Un gestor de contenidos o CMS (del inglés, Content Management System) es una aplicación web o software que permite crear, administrar y publicar una página web en Internet.

Estas aplicaciones tienen un panel de administración, también llamada «interfaz», que permite crear, editar o publicar contenido web. Los gestores de contenido normalmente poseen plugins, extensiones o complementos que permiten añadir o modificar funcionalidades en el sitio. Inicialmente los gestores de contenidos o CMS nacieron para facilitarles un poco el trabajo a los webmasters, pero han avanzado tanto en los últimos años que ahora casi cualquier persona es capaz de montar su propia web sin tener que externalizar el servicio o contratar a un profesional (Acibeiro, 2023).

Drupal v9.5.10 es un sistema de gestión de contenidos multipropósito, modular, libre y con una amplia capacidad de personalización. Permite publicar archivos, imágenes, artículos, al igual que crear y administrar todo tipo de contenidos como votaciones, encuestas, foros, entre otros. A pesar de no ser tan popular como WordPress, Drupal es uno de los CMS más completos para grandes portales corporativos, además se trata de una plataforma flexible y fácilmente integrable con otras soluciones de negocio. Se

emplea Drupal porque da la oportunidad de crear etiquetas, categorías, taxonomías y personalizar. Asimismo, es posible instalar plugins, extensiones y editar el código fuente al antojo.

1.8.2.7. Framework de desarrollo

Un marco de trabajo (framework) es una gran librería o conjunto de librerías, donde además de facilitar funciones para su uso, dispone de una sintaxis o metalenguaje específico del marco de trabajo y una forma de organización de su código. Para usar un marco de trabajo no basta con conocer el nombre de la función a utilizar: hay que saber qué sintaxis emplea, qué obligaciones impone (a la hora de organizar el código, archivos) y su lógica o filosofía de trabajo. El empleo de marcos de trabajo se ha extendido debido a la gran complejidad de las aplicaciones que se desarrollan, ya que facilitan su organización y mantenimiento (Lucena, 2023).

Bootstrap v4 es un framework front-end utilizado para desarrollar aplicaciones web y sitios mobile first, o sea, con un layout que se adapta a la pantalla del dispositivo utilizado por el usuario. El framework combina CSS y JavaScript para estilizar los elementos de una página HTML. Permite mucho más que, simplemente, cambiar el color de los botones y los enlaces.

Esta es una herramienta que proporciona interactividad en la página, por lo que ofrece una serie de componentes que facilitan la comunicación con el usuario, como menús de navegación, controles de página, barras de progreso y más (Bootstrap, 2020).

Symfony v4 es un entorno de trabajo estandarizado (framework PHP) que se utiliza para el desarrollo de aplicaciones web y es de los más utilizados en el entorno de desarrolladores de apps. En otras palabras, es una herramienta para desarrolladores para crear aplicaciones en PHP. La estructura Modelo Vista Controlador del framework permite un progreso en función de las necesidades de los clientes, además tiene una gran cantidad de plantillas para desarrollar webs (Qualitydevs, 2019).

1.8.2.8. Motor de plantillas para el lenguaje de programación PHP

Twig v2 es un motor de plantillas desarrollado para el lenguaje de programación PHP. Nace con el objetivo de facilitar a los desarrolladores de aplicaciones web el trabajo con la parte de las vistas, gracias a que se trata de un sistema que resulta muy sencillo de aprender y capaz de generar plantillas con un código preciso y fácil de leer. Actualmente el código se distribuye bajo licencia BSD y es utilizado por el framework Symfony.

Cuando nos referimos al lenguaje PHP, una de las plantillas más utilizadas es una plantilla PHP, en la que se mezcla texto interpretado por PHP y en el que se mezclan etiquetas HTML y código PHP para formar la vista que verá el usuario.

1.8.2.9. Biblioteca de JavaScript

jQuery es una de las bibliotecas JavaScript más populares y ampliamente usadas. Su objetivo fundamental es simplificar algunas tareas comunes, como buscar elementos en el DOM, manipular esos elementos, editar atributos HTML y propiedades CSS, definir manejadores de eventos, entre otras. Cuenta además con utilidades Ajax para realizar peticiones HTML dinámicamente, así como funciones de propósito general para trabajar con objetos y colecciones.

1.9. Conclusiones del capítulo

El análisis de los principales conceptos asociados al dominio del problema facilitó una mejor comprensión del presente trabajo. El estudio del estado del arte realizado, permitió obtener las tendencias de la forma de estructurar y organizar los sistemas para la gestión de información en el proceso de entrega y devolución de bibliografía física, proporcionando ideas para el diseño del sistema. Arrojó resultados tales como la generalización del empleo de las tecnologías web para la implementación de este tipo de sistemas y la independencia con la que deben cumplir, al no estar ligados a sistemas superiores. Este estudio demostró que no existen soluciones alternativas que aplicar a la problemática presente. Por otra parte, se realizó la selección de los lenguajes, herramientas y metodología para guiar el proceso de desarrollo de software, lo que permitirá lograr mayor productividad, mejores resultados y garantizará la obtención de un producto de mayor calidad que satisfaga plenamente las necesidades del cliente.

Capítulo 2

Características del sistema y propuesta de solución

2.1. Introducción

El presente capítulo se centra en brindar una propuesta de solución basándose en el análisis del proceso de entrega y devolución de libros en la Facultad 2 de la UCI. Se realiza una descripción detallada de dicho proceso y se modela el mismo haciendo uso de Business Process Model and Notation (Notación para el Modelado de Procesos de Negocio, BPMN). Se plantea la estructura modular del sistema y los roles que se definen para su explotación. Se describen además las listas de funcionalidades y propiedades del sistema para una mejor comprensión y dar cumplimiento a los objetivos planteados. Se realizan las historias de usuario, plan de iteraciones, tarjetas CRC y demás artefactos exigidos por la Metodología XP y sus fases.

Fase 1: Exploración

Durante la fase de exploración, el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología (Molina Montero, Vite Cevallos, & Dávila Cuesta, 2018).

Como se puedo apreciar en el capítulo anterior se realizó un análisis de la temática a partir de la investigación realizada mediante la entrevista a los vicedecanos de administración de cada una de las facultades de la Universidad. Además, se realizó un estudio de sistemas homólogos con el fin de recopilar información de características necesarias y buenas prácticas para la confección de la aplicación web, para lo cual fueron seleccionadas las herramientas estudiadas durante el transcurso de la carrera.

2.2. Proceso de entrega y devolución de bibliografía física docente en la Facultad 2 de la UCI

Actualmente el proceso puede ser dividido en dos subprocesos: entrega de bibliografía física y devolución de bibliografía física.

El grupo de trabajo está compuesto por un supervisor del departamento el cual atiende las solicitudes de entrega y devolución de libros por parte de los clientes. Los clientes pueden ser trabajadores, profesores y estudiantes de la Facultad 2 de la UCI.

El consumo de los servicios por los clientes comienza con la realización de una solicitud de forma presencial en el almacén de libros de la Facultad 2 de la UCI. Actualmente, la solicitud es recibida por el supervisor del departamento quien realiza la tarea de registrar los datos. Después se procede a notificar al cliente y entregar o recoger la bibliografía física. La recepción de las solicitudes, el registro de la información y la entrega y devolución de bibliografía física, son actividades realizadas manualmente.

Estas labores generan una gran cantidad de información, la cual queda plasmada en forma de documentos digitales tales como tablas de Excel y documentos de formato duro. A continuación, se realiza la descripción detallada de cada subproceso con su respectivo modelado haciendo uso de BPMN.

2.2.1. Subproceso entrega de bibliografía física

El proceso comienza con una solicitud que se realiza por parte del cliente de forma presencial en el almacén de la Facultad 2 de la UCI, la cual es recibida y registrada por el supervisor, quien verifica la identidad del cliente y crea la ficha de entrega. Si ha extraído libros con anterioridad, el supervisor procede a actualizar la ficha de entrega, sin necesidad de crear una nueva y verifica que los libros extraídos hayan sido entregados, en caso contrario, lo notifica al cliente.

Luego valora a través de una consulta, la disponibilidad de los recursos bibliográficos con que cuenta el departamento. La solicitud puede ser atendida o no, de no ser posible se pospone la realización de la solicitud hasta que se cuente con los recursos necesarios. En caso de que se pueda realizar, el supervisor se encarga de entregar la bibliografía solicitada. Una vez concluidas estas tareas, el supervisor notifica al cliente que la solicitud ha sido registrada. En la Figura 1 se puede apreciar este proceso con mayor detalle.

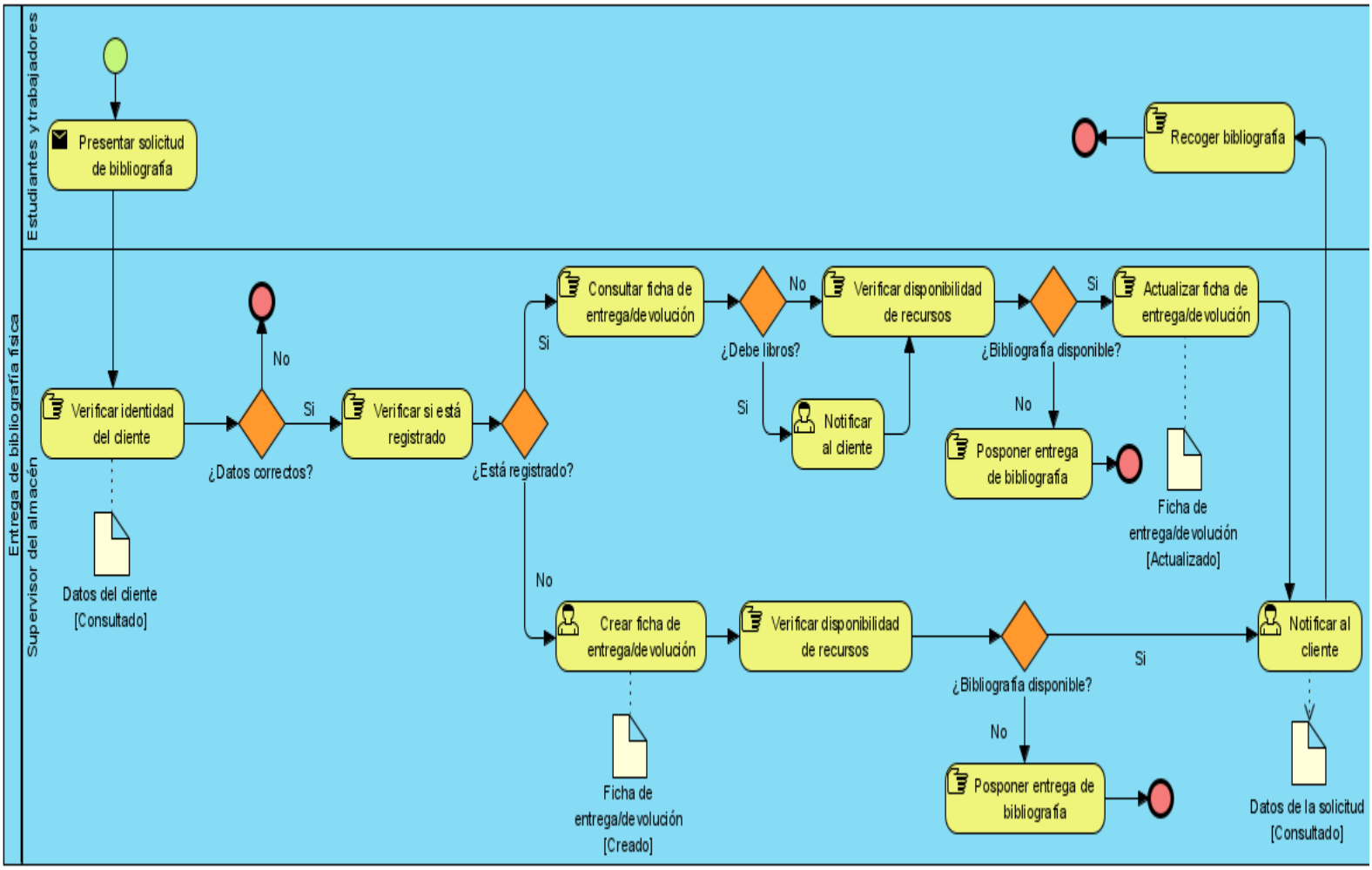


Figura 1: Subproceso entrega de bibliografía física

2.2.2. Subproceso devolución de bibliografía física

El proceso comienza con una solicitud presentada por el cliente de forma presencial en el almacén de la Facultad 2 de la UCI. La solicitud es recibida por el supervisor, quien verifica la identidad del cliente, revisa si debe otros libros y en caso positivo, lo notifica al cliente.

El supervisor se encarga de recoger la bibliografía y actualizar la ficha de devolución. Una vez concluidas estas tareas, el supervisor notifica al cliente que la devolución ha sido registrada. Luego, en caso de ser estudiante y pertenecer a un año terminal, el supervisor notifica a la Vicedecana de Formación, la cual procede a firmar el documento andar UCI del estudiante. En la siguiente figura (**Figura 2**) se puede apreciar este proceso con mayor detalle.

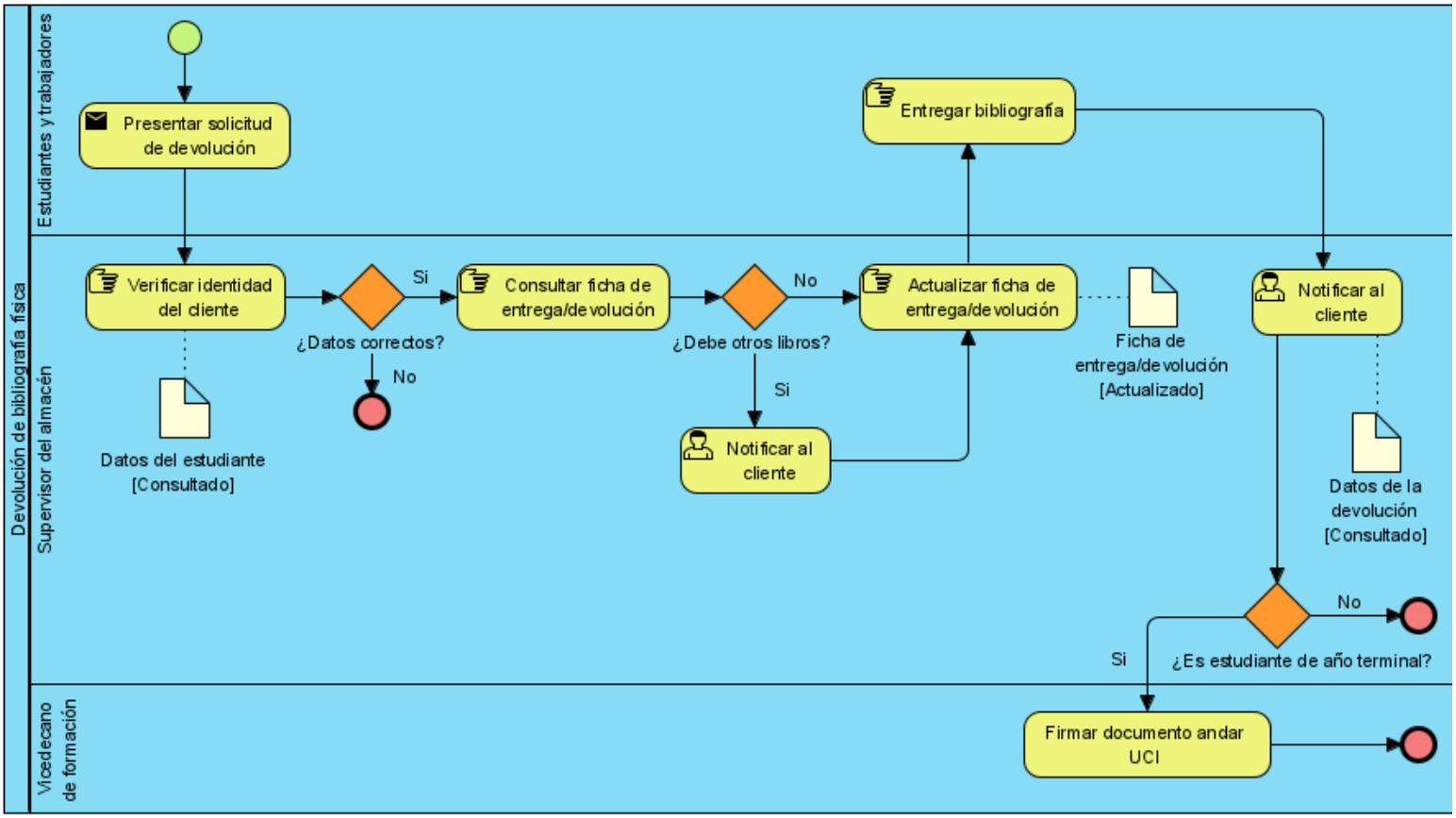


Figura 2: Subproceso devolución de bibliografía física

2.3. Historias de usuario

Las historias de usuario (HU) son la forma en que se especifican en XP los requisitos del sistema. Estas se escriben desde la perspectiva del cliente, aunque los desarrolladores pueden brindar también su ayuda en la identificación de las mismas (Beck, 2000).

Una HU es una explicación general e informal de una función de software escrita desde la perspectiva del usuario final o cliente. Su propósito es articular cómo proporcionará una función de software al cliente. Se trata de tarjetas de papel en las cuales el cliente describe las características que el sistema debe poseer, sean requisitos funcionales o no. El tratamiento de las historias de usuario es dinámico y flexible. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas (Beck, 2000).

Durante la fase de exploración se identificaron las siguientes historias de usuario:

- Autenticar usuario.
- Gestionar usuarios.
- Gestionar libros.
- Gestionar solicitudes.
- Generar inventario.

Las HU definidas se clasificaron atendiendo a diferentes criterios, facilitando así su organización y la correcta planificación del proyecto. Los parámetros definidos en cada una de las HU se detallan a continuación.

- Número y nombre: Para identificar la HU.
- Prioridad:
    - Alta: Constituyen funcionalidades principales del sistema, o forman parte esencial de la arquitectura del mismo.
    - Media: Son funcionalidades importantes y de gran valor para el usuario pero que no impiden poner el proyecto en marcha si no se tienen.
    - Baja: Son funcionalidades que sería deseable tener y podrían incluirse en caso de que hubiese recursos para ello.
- Riesgo en desarrollo:
    - Alto: En caso de tener algún error de implementación, pueden afectar la disponibilidad del sistema.
    - Medio: En caso de presentar errores retrasan la entrega de la versión.

    - Bajo: En caso de presentar errores, estos pueden ser tratados con facilidad y no afectan el desarrollo del proyecto.
- Iteración asignada: Número de la iteración a la que ha sido asignada.
- Puntos estimados: Tiempo estimado de desarrollo para completar la HU. Un punto equivale a una semana ideal de programación.
- Descripción: Es donde se define la funcionalidad que se quiere desarrollar.
- Observaciones: Detalles a tener en cuenta para desarrollar la HU correctamente.

Las HU del presente proyecto fueron elaboradas siguiendo el formato propuesto por Mike Cohn en su libro User Stories Applied (Historias de Usuario Aplicadas) (Cohn, 2004). A continuación, se muestra la historia de usuario definidas.

Tabla 2: Historia de usuario "Sistema de autenticación"

Historia de Usuario	
Numero: 1	**Nombre:** Autenticar usuario
Programador: Yadian Tresgallos González	
Prioridad: Alta	**Riesgo en desarrollo:** Medio
Iteración asignada: 1	**Puntos estimados:** 1
Descripción: Un usuario puede acceder al sistema con su nombre de usuario y contraseña.	
Observaciones: Un usuario puede acceder al sistema, siempre y cuando este registrado en el sistema, o ya se haya autenticado anteriormente.	
Prototipo de interfaz:	

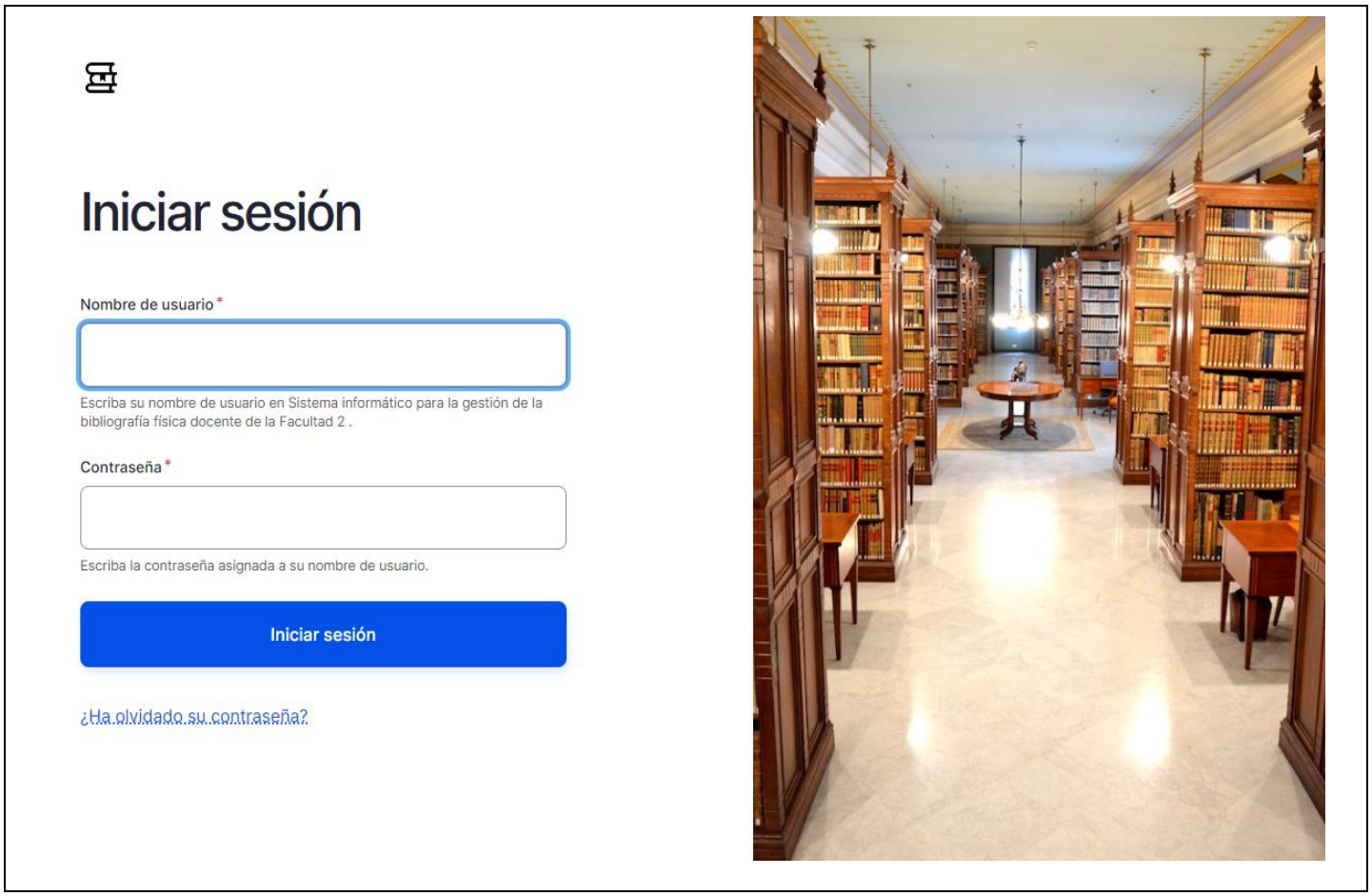


Tabla 3: Historia de usuario "Gestionar usuarios"

Historia de Usuario	
Numero: 2	**Nombre:** Gestionar usuarios
Programador: Yadian Tresgallos González	
Prioridad: Media	**Riesgo en desarrollo:** Media
Iteración asignada: 1	**Puntos estimados:** 2
Descripción: El administrador puede gestionar (crear, modificar, eliminar, listar y filtrar) a los usuarios que emplean los servicios del sistema.	
Observaciones: Los usuarios pueden modificar los datos de su perfil (dirección de correo electrónico, nombre, contraseña, imagen).	
Prototipo de interfaz:	

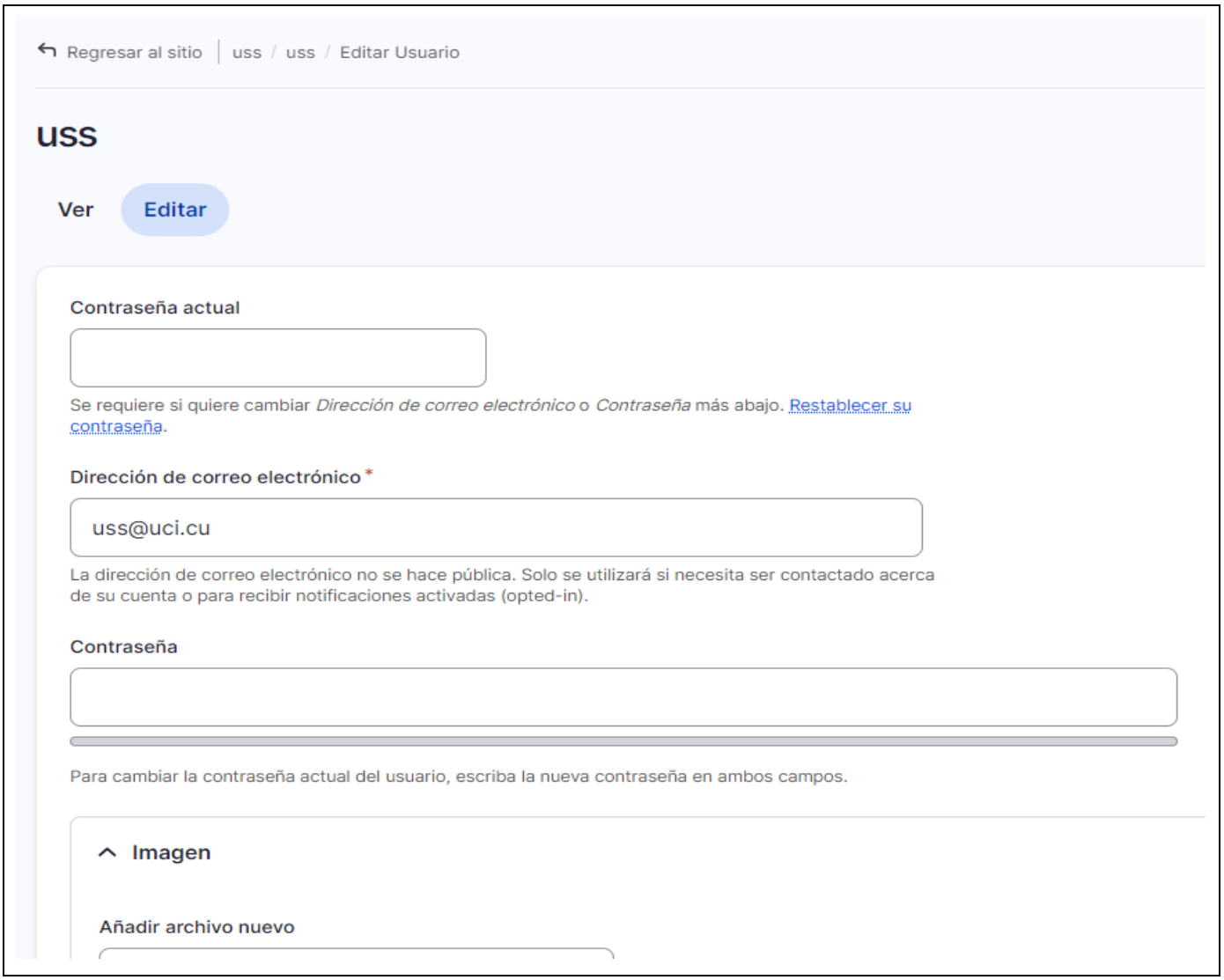


Tabla 4: Historia de usuario "Gestionar libros"

Historia de Usuario	
Numero: 3	**Nombre:** Gestionar libros
Programador: Yadian Tresgallos González	
Prioridad: Alta	**Riesgo en desarrollo:** Alta
Iteración asignada: 2	**Puntos estimados:** 2
Descripción: Los usuarios pueden gestionar (añadir, modificar, eliminar, listar y filtrar) los libros disponibles en el almacén de la facultad.	
Observaciones: -	
Prototipo de interfaz:	

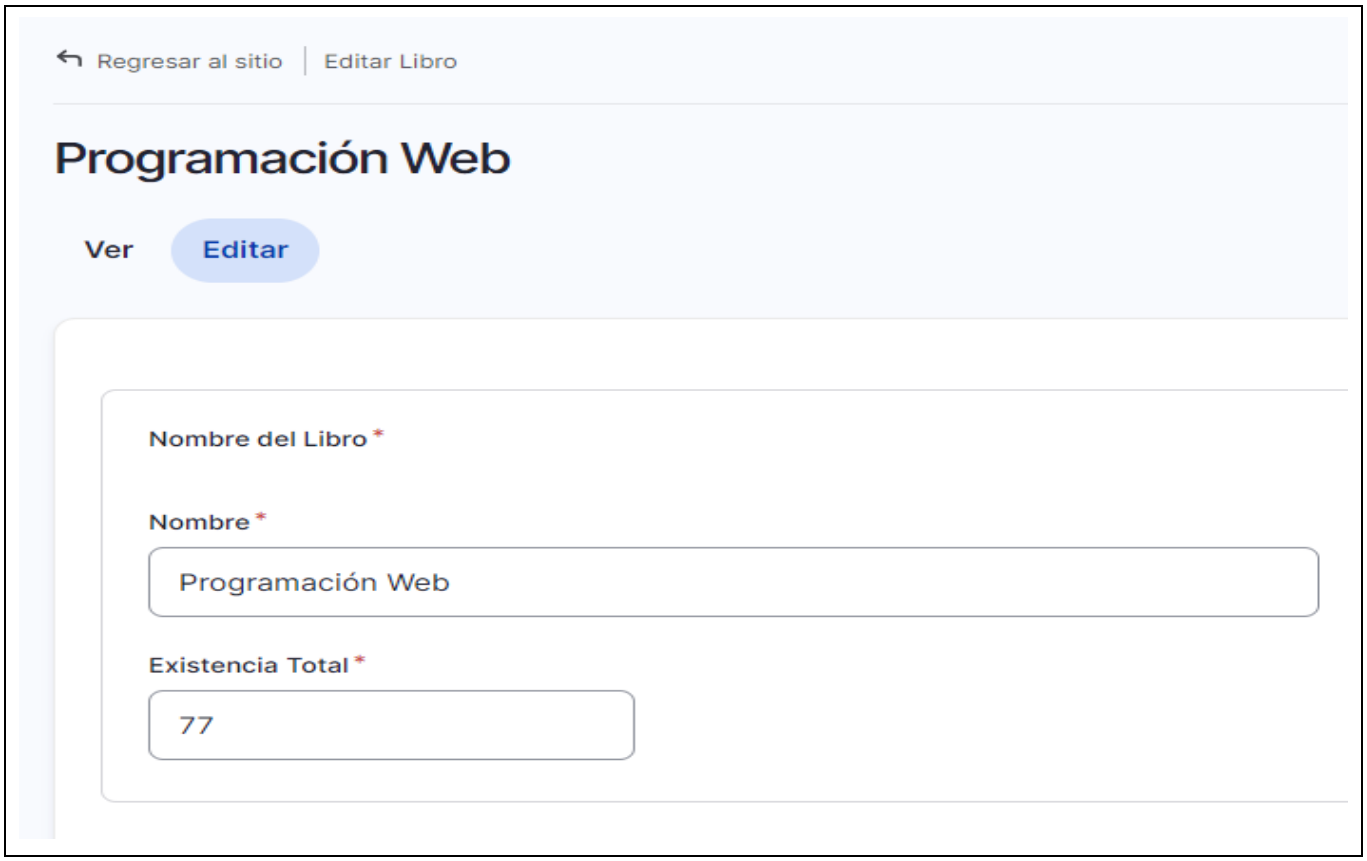


Tabla 5: Historia de usuario "Gestionar solicitudes"

Historia de Usuario	
Numero: 4	**Nombre:** Gestionar solicitudes
Programador: Yadian Tresgallos González	
Prioridad: Alta	**Riesgo en desarrollo:** Alta
Iteración asignada: 2	**Puntos estimados:** 2
Descripción: Los usuarios pueden gestionar (crear, modificar, eliminar, listar y filtrar) las solicitudes de entrega y devolución de textos de la facultad.	
Observaciones: -	
Prototipo de interfaz:	

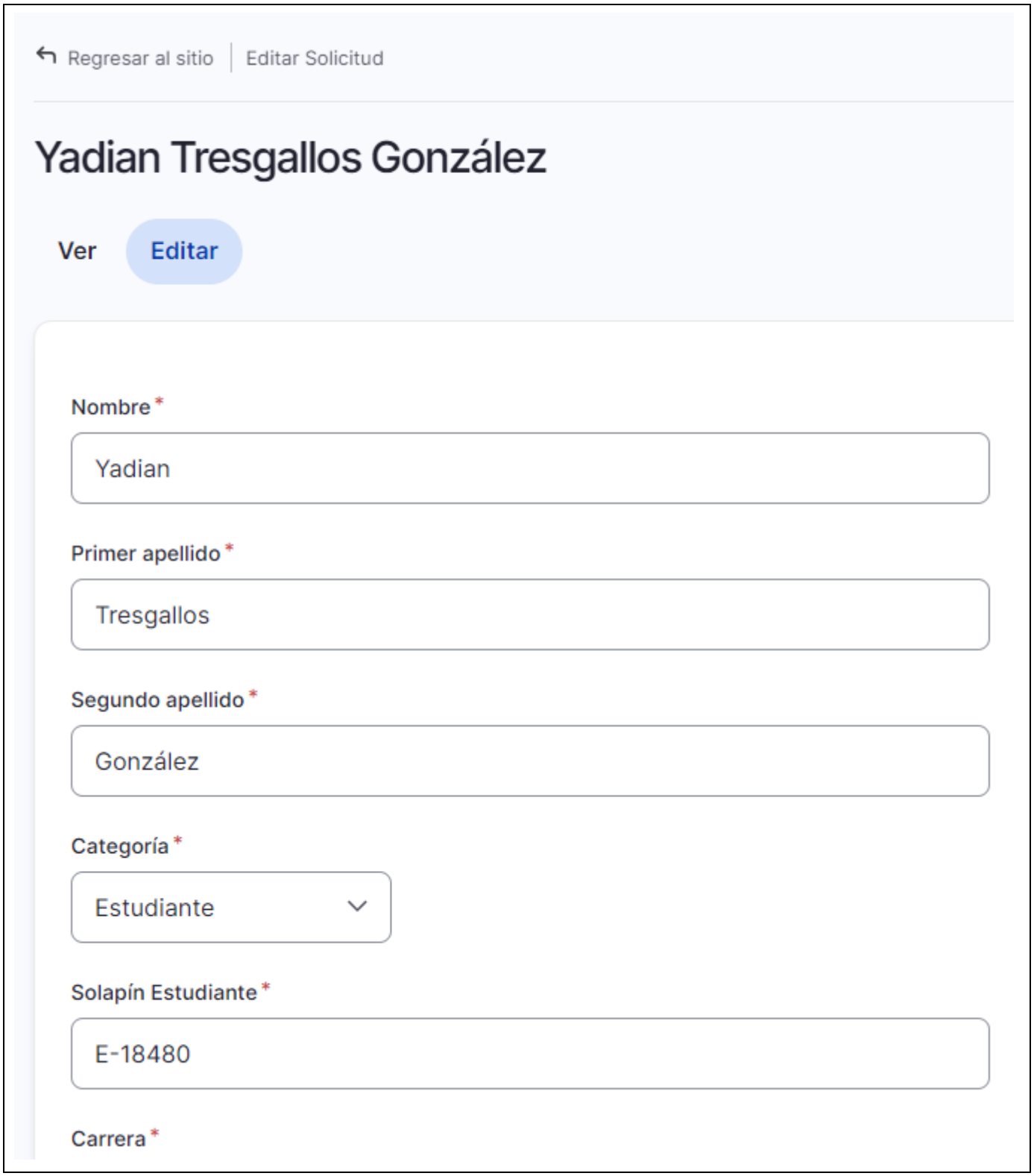
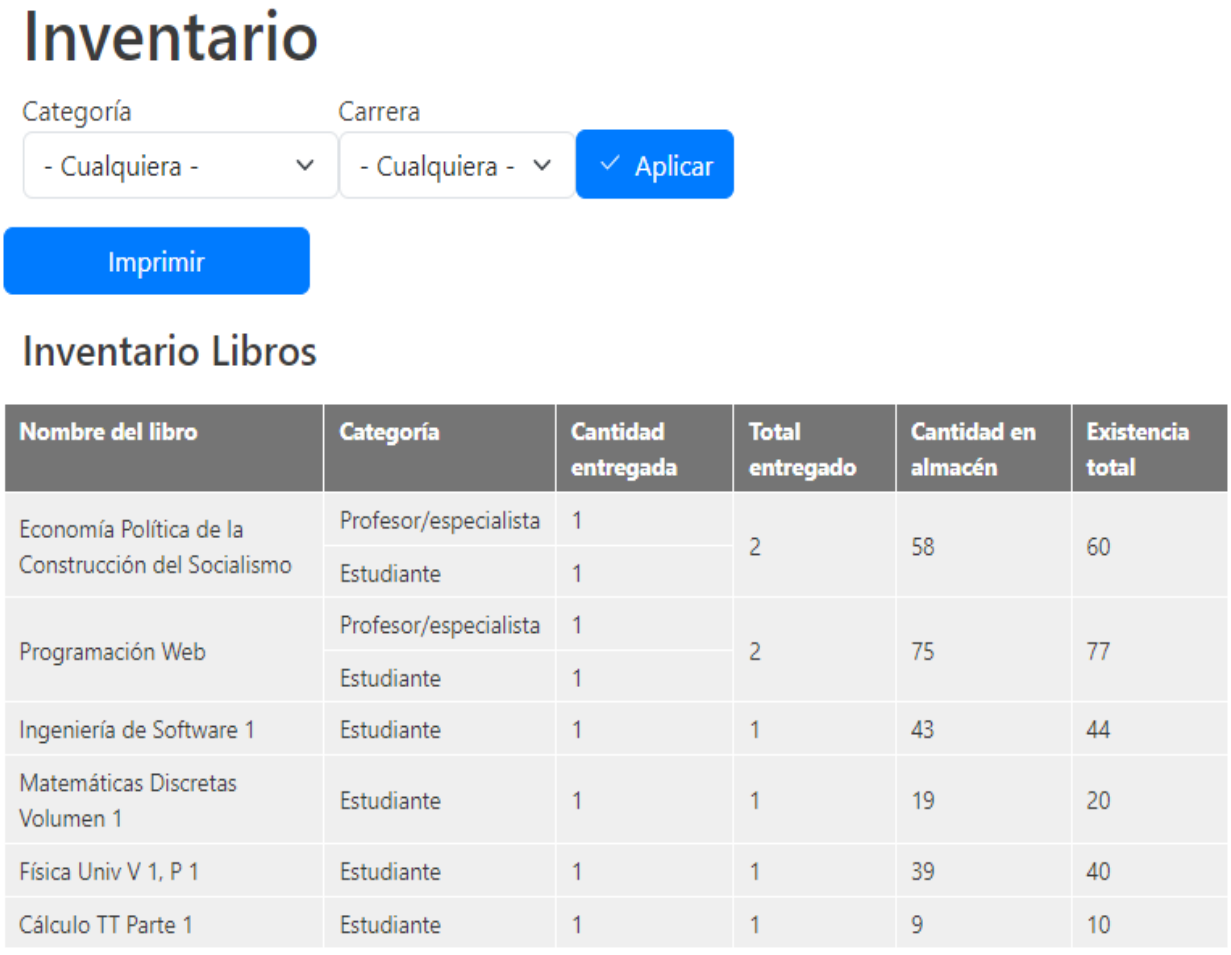


Tabla 6: Historia de usuario "Generar inventario"

Historia de Usuario	
Numero: 5	**Nombre:** Generar inventario

Programador: Yadian Tresgallos González	
Prioridad: Media	**Riesgo en desarrollo:** Media
Iteración asignada: 3	**Puntos estimados:** 1
Descripción: Los usuarios pueden acceder a un inventario en formato de tabla que muestra un listado de la cantidad total, cantidad de entregados y cantidad de libros existentes en el almacén. Además, permite imprimir y/o generar un documento (formato PDF) de estas estadísticas.	
Observaciones: -	

Prototipo de interfaz:



Nombre del libro	Categoría	Cantidad entregada	Total entregado	Cantidad en almacén	Existencia total
Economía Política de la Construcción del Socialismo	Profesor/especialista	1	2	58	60
	Estudiante	1			
Programación Web	Profesor/especialista	1	2	75	77
	Estudiante	1			
Ingeniería de Software 1	Estudiante	1	1	43	44
Matemáticas Discretas Volumen 1	Estudiante	1	1	19	20
Física Univ V 1, P 1	Estudiante	1	1	39	40
Cálculo TT Parte 1	Estudiante	1	1	9	10

Fase 2: Planificación

2.4. Planificación de entregas

Se cumple con la segunda fase de la Metodología XP donde el cliente establece la prioridad de cada historia de usuario. En correspondencia, los programadores estiman el esfuerzo necesario para la realización de cada una de estas. Se toman en cuenta acuerdos sobre el contenido de la primera entrega determinada y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres semanas.

2.4.1. Estimación de tiempo por historias de usuario

La programación utilizando la metodología XP basa sus procesos de planificación en estimaciones temporales de las historias de usuario, las cuales deben ser realizadas por los desarrolladores durante las diversas reuniones de planificación.

Una historia de usuario debe ser lo suficientemente pequeña como para que el equipo la desarrolle durante una entrega de una a tres semanas; más de tres semanas implica que se le debe señalar al cliente, para que divida una historia de usuario y menos de una semana implica que la historia es demasiado sencilla y por lo que se deben unir dos o más de ellas para su mejor interpretación (Beck, 2000).

Las estimaciones de esfuerzo asociado a la implementación de las historias la establecen los programadores utilizando como medida el punto de estimación. Un punto de estimación, equivale a una semana ideal de programación. Las historias generalmente valen de 1 a 3 puntos de estimación. Por otra parte, el equipo de desarrollo mantiene un registro de la “velocidad” de desarrollo, establecida en puntos de iteración, basándose principalmente en la suma de puntos correspondientes a las historias de usuario que fueron terminadas en la última iteración.

En la siguiente tabla de estimación del tiempo de las historias de usuario, se usó la denotación de punto de estimación por semana.

Tabla 7: Estimación de tiempo

Denotación de punto de estimación	**Interpretación**
1 punto	1 semana

Tabla 8: Estimación de tiempo por HU

Historia de usuario	**Punto estimado**
Autenticar usuario	1
Gestionar usuarios	2

Gestionar libros	2
Gestionar solicitudes	2
Generar inventario	1

Las estimaciones permitieron confeccionar una evaluación puntual del tiempo de implementación de cada historia de usuario para la posterior elaboración del plan de iteración. Una vez realizadas las estimaciones fue preciso construir un plan de iteración donde se pueden agrupar estas historias y dar cumplimento de manera paulatina.

2.4.2. Plan de iteraciones

Es la práctica en donde el equipo establece el rumbo cada un par de semanas. Los equipos construyen software en iteraciones de dos semanas, y entregan software útil al finalizar cada iteración. Los programadores las descomponen en tareas, y estiman su costo (a un nivel de detalle más fino que durante la Planificación de la Entrega). El equipo entonces se compromete a terminar ciertas características basándose en la cantidad de trabajo que pudieron terminar en la iteración anterior.

El contenido de la iteración puede variar, dependiendo de la posición del ciclo de vida de la naturaleza del proyecto. El plan de iteración incluye proporciones relevantes a todas las disciplinas para cada iteración en particular, la prioridad de implementación se evalúa en base al cliente y el equipo de desarrollo, el impacto del riesgo en base al juicio de experto. Los planes de iteración por lo general muestran un planeamiento detallado de quién va a realizar una tarea/actividad de acuerdo en conformidad a que criterios de evaluación.

Tabla 9: Plan de iteraciones

Iteración	**Historia de usuario**	**Prioridad**	**Esfuerzo estimado**	
1	Autenticar usuario	Alta	1	3
	Gestionar usuarios	Media	2	
2	Gestionar libros	Alta	2	4
	Gestionar solicitudes	Alta	2	
3	Generar inventario	Media	1	1

Una vez realizado el plan de iteración se pudo agrupar las diferentes historias de usuario en iteraciones teniendo en cuenta las características que rigen la metodología XP. Con este plan de iteraciones ya es posible realizar un plan de entrega que será entregado al cliente y que el grupo de desarrollo está obligado a hacer cumplir.

2.4.3. Plan de entrega

Basándonos en las historias de usuario definidas para el desarrollo del sistema web, se ha elaborado el siguiente plan de entrega, el cual muestra las historias de usuario que se llevarán a cabo en cada iteración.

Tabla 10: Plan de entrega

	Iteración 1	**Iteración 2**	**Iteración 3**
Cantidad de HU	2	2	1
Fecha de inicio	7/08/2023	28/08/2023	25/09/2023
Fecha de entrega	25/08/2023	22/09/2023	29/09/2023

Fase 3: Diseño

Teniendo en cuenta lo que plantea la metodología XP, en esta fase se confeccionan las tarjetas Clase Responsabilidad-Colaborador para la descripción de las principales clases de los módulos desarrollados. Se define la arquitectura del sistema y los estándares de codificación, así como los patrones de diseño utilizados en el desarrollo de la propuesta.

2.5. Patrón Arquitectónico

Los patrones arquitectónicos se utilizan para expresar una estructura de organización base o esquema para un software. Proporcionan un conjunto de subsistemas predefinidos, especificando sus responsabilidades, reglas y directrices que determinan la organización, comunicación, interacción y relaciones entre ellos. Los patrones arquitectónicos heredan mucha de la terminología y conceptos de patrones de diseño, pero se centran en proporcionar modelos y métodos reutilizables específicamente para la arquitectura general de los sistemas de información (Gomez Mejia, 2013).

El sistema será desarrollado siguiendo el patrón clásico de arquitectura Modelo Vista Controlador (MVC), el cual separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

Modelo: Tiene la responsabilidad de relacionar los datos con los que la aplicación va a operar. A esto se le llama lógica de negocio, es decir, es la parte de la aplicación que se encarga de mapear las actividades del mundo real a la forma a la que se va a modificar la información.

Vista: Parte de la aplicación que contiene la interfaz gráfica. Cada elemento gráfico que interactúa con el usuario es la vista y su función es la de ser la capa que obtiene información sobre lo que quiere el usuario, a lo cual vamos a llamar eventos, la vista muestra siempre la información del modelo.

Controlador: Responde a eventos o acciones que realice el usuario a través de la vista para poder solicitar una operación de información, también es responsable de mostrarle al usuario la vista adecuada dependiendo de la solicitud recibida, por lo cual es el vínculo que une al modelo con la vista.

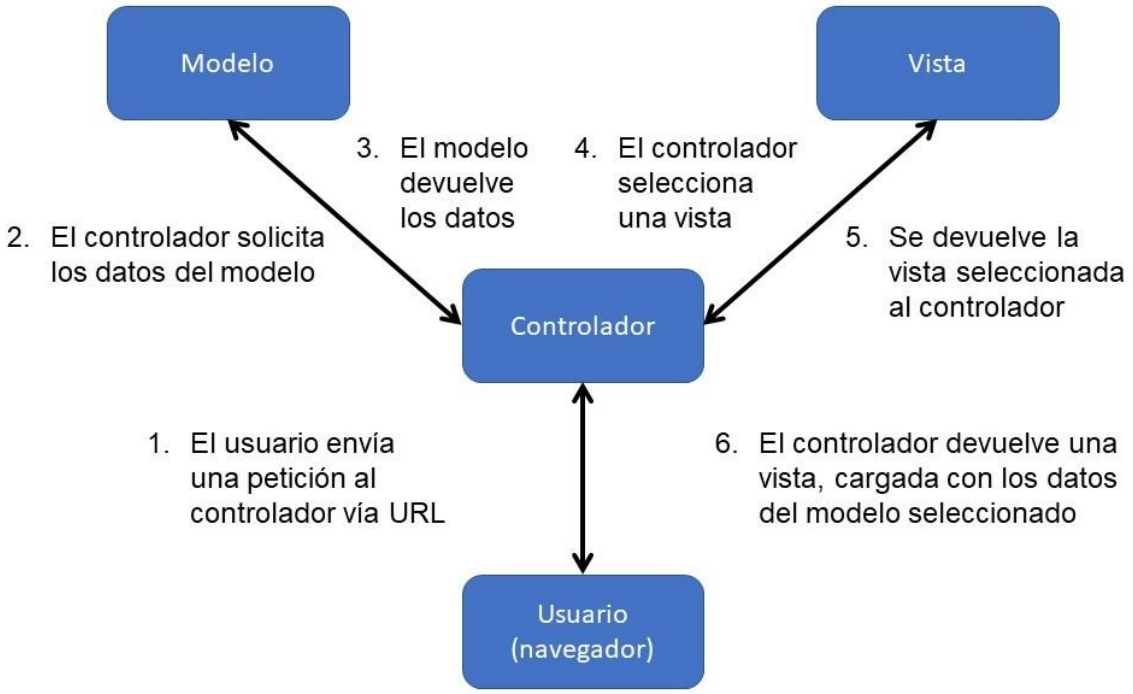


Figura 3: Funcionamiento del patrón Modelo Vista Controlador

En resumen, la vista muestra el modelo al usuario, este activa eventos en las vistas los cuales son recibidos por el controlador. El controlador manipula al modelo dependiendo de la solicitud. El modelo devuelve un conjunto de datos acordes a la solicitud realizada por el controlador, para que finalmente el controlador muestre una nueva vista al usuario con la información actualizada del modelo.

2.6. Patrones de diseño

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software. Muchos patrones proporcionan guías en el modo en que deberían asignarse las responsabilidades a los objetos, dada una categoría específica del problema (Larman, 2005).

2.6.1. Patrones GoF

Los Patrones de Diseño GoF (Gang of Four) son un conjunto de patrones de diseño de software que fueron definidos por cuatro autores reconocidos en el libro: "Design Patterns: Elements of Reusable Object-Oriented Software". Estos cuatro autores son Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides, también conocidos como la "Pandilla de cuatro" (GoF).

Los patrones de diseño GoF proporcionan un vocabulario y una guía común para ayudar a los diseñadores y desarrolladores a comunicarse sobre soluciones de diseño probadas y bien estructuradas. Al aplicar estos patrones, se busca mejorar la eficiencia, la flexibilidad y la reusabilidad del software.

Los patrones de diseño GoF se organizan en tres categorías:

- Patrones de creación.
- Patrones de estructura.
- Patrones de comportamiento.

Creacionales: Los patrones de creación proporcionan diversos mecanismos de creación de objetos, que aumentan la flexibilidad y la reutilización del código existente de una manera adecuada. Esto le da al programa más flexibilidad para decidir qué objetos deben crearse para un caso de uso dado (Martínez Canelo, 2020).

- **Singleton (Único):**

Se utiliza para asegurar que una clase tenga exactamente una instancia y proporcionar un punto de acceso global a ella. Resulta útil cuando exactamente un objeto es necesario para coordinar acciones en todo el sistema y en situaciones donde se necesite que solo haya una instancia de una clase, como una conexión a una base de datos o un archivo de registro (Martínez Canelo, 2020).

En el siguiente fragmento de código (**Figura 4**), Database::getConnection() es un método que devuelve la única instancia de la conexión a la base de datos, garantizando que solo exista una conexión a la base de datos.

```php
public function createDatabase($database) {
  // Escape the database name.
  $database = Database::getConnection()->escapeDatabase($database);

  try {
    // Create the database and set it as active.
    $this->connection->exec("CREATE DATABASE $database");
    $this->connection->exec("USE $database");
  }
  catch (\Exception $e) {
    throw new DatabaseNotFoundException($e->getMessage());
  }
}
```

Figura 4: Ejemplo de empleo del patrón Singleton

Estructurales: Facilitan soluciones y estándares eficientes con respecto a las composiciones de clase y las estructuras de objetos. El concepto de herencia se utiliza para componer interfaces y definir formas de componer objetos para obtener nuevas funcionalidades (Martínez Canelo, 2020).

- **Decorator (Decorador):**

Permite añadir comportamientos a un objeto de manera dinámica, sin afectar el comportamiento de otros objetos de la misma clase. Este patrón es útil a la hora de dividir la funcionalidad entre clases con áreas de preocupación únicas, así como para extender la funcionalidad de una clase sin modificarla (Martínez Canelo, 2020).

En el siguiente ejemplo (**Figura 5**), RouteCacheContext es un decorador que agrega el contexto de caché de la ruta actual a la pila de caché. Cuando se llama al método getContext, se obtiene el nombre de la ruta actual y se devuelve. Si no hay ninguna ruta actual, se devuelve la cadena 'none'.

```php
namespace Drupal\Core\Cache\Context;

use Drupal\Core\Cache\CacheableDependencyInterface;

class RouteCacheContext implements CacheableContextInterface {

 /**
  * {@inheritdoc}
  */
 public static function getLabel() {
  return t('Route');
 }

 /**
  * {@inheritdoc}
  */
 public function getContext() {
  $route_name = \Drupal::routeMatch()->getRouteName();
  return $route_name ?: 'none';
 }

}
```

Figura 5: Ejemplo de empleo del patrón Decorator

Este decorador se utiliza automáticamente por Drupal cuando se habilita el caché. De esta manera, Drupal puede almacenar en caché, diferentes versiones de una página web dependiendo de la ruta actual. Por ejemplo, se puede tener una versión en caché de la página de inicio para los usuarios no autenticados y otra versión en caché para los usuarios autenticados.

- **Bridge (Puente):**

Permite separar una gran clase o un conjunto de clases estrechamente relacionadas en dos jerarquías independientes: la abstracción y la implementación. Esto significa que se puede desarrollar estas dos jerarquías de manera independiente. Se utiliza para desacoplar una abstracción de su implementación para que ambas puedan variar independientemente. Esto significa que se puede crear una interfaz de puente que utilice los principios de la programación orientada a objetos para separar las responsabilidades en diferentes clases abstractas (Martínez Canelo, 2020).

La capa de abstracción de la base de datos de Drupal se implementa de manera similar al patrón de diseño de Bridge. Los módulos deben escribirse de forma independiente del sistema de base de datos que se utiliza, y la capa de abstracción lo proporciona.

En el siguiente fragmento de código (**Figura 6**), la interfaz CacheBackendInterface actúa como el "puente" entre la abstracción (la necesidad de almacenar datos en caché) y las implementaciones concretas (cómo se almacenan los datos en caché). Las implementaciones concretas MemoryCacheBackend y FileCacheBackend representan diferentes formas de almacenar los datos en caché. La clase CacheFactory es la entidad que necesita obtener datos en caché y tiene una relación con un objeto de la interfaz CacheBackendInterface. Cuando se llama al método get de la clase CacheFactory, se llama al método get del objeto CacheBackendInterface. Esto permite que la clase CacheFactory sea independiente de los detalles de cómo se almacenan los datos en caché, y permite que el sistema de caché sea cambiado o extendido sin afectar la clase.

```php
interface CacheBackendInterface {
 public function get($cid, $allow_invalid = FALSE);
 public function set($cid, $data, $expire = Cache::PERMANENT, array
 // ...
}

// Implementación concreta MemoryCacheBackend
class MemoryCacheBackend implements CacheBackendInterface {
 // Implementación de los métodos de la interfaz
}

// Implementación concreta FileCacheBackend
class FileCacheBackend implements CacheBackendInterface {
 // Implementación de los métodos de la interfaz
}

// Clase CacheFactory
class CacheFactory {
 protected $backend;

 public function __construct(CacheBackendInterface $backend) {
 $this->backend = $backend;
 }

 public function get($cid) {
 // Utiliza el backend para obtener los datos en caché
 return $this->backend->get($cid);
 }
```

Figura 6: Ejemplo de empleo del patrón Bridge

Comportamiento: Se ocupa de la comunicación entre objetos de clase. Se utilizan para detectar la presencia de patrones de comunicación existentes y manipularlos. Estos patrones de diseño están específicamente relacionados con la comunicación entre objetos (Martínez Canelo, 2020).

- **Observer (Observador):**

Permite a un objeto, llamado sujeto, mantener una lista de sus dependientes, llamados observadores, y notificarlos automáticamente sobre cualquier cambio de estado. Es útil cuando se necesita mantener una relación de uno a muchos entre los objetos, donde un cambio en un objeto debe notificar a todos sus dependientes (Martínez Canelo, 2020).

El patrón de Observador es generalizado en todo Drupal, ya que muchos de los ganchos de Drupal esencialmente permiten que los módulos se registren como observadores de los objetos de Drupal. Por ejemplo, cuando se realiza una modificación a un vocabulario en el sistema de taxonomía de Drupal, se llama a un gancho de taxonomía en todos los módulos que lo implementan. Al implementar el gancho, los módulos se han registrado como observadores del objeto de vocabulario; cualquier cambio en él se puede actuar según corresponda (Drupal, 2023).

En el siguiente ejemplo (**Figura 7**), se utiliza el módulo Paragraph el cual proporciona una interfaz de usuario para crear y administrar contenido de tipo "Paragraph". En este caso paragraphs es el observador y el gancho hook_entity_insert es el sujeto. Cuando se inserta una nueva entidad, el gancho hook_entity_insert notifica al observador paragraphs y este puede reaccionar al cambio de estado.

```php
/**
 * Implements hook_ENTITY_TYPE_insert().
 */
function paragraphs_node_insert(
   Drupal\node\NodeInterface $node
) {
   // Código para ejecutar cuando se inserta una nueva entidad.
   drupal_set_message(
       t(
           'Se ha insertado una nueva entidad: @title',
           ['@title' => $node->getTitle()]
       )
   );
}
```

Figura 7: Ejemplo de empleo del patrón Observer

- **Chain of responsibility (Cadena de responsabilidad):**

Permite encadenar una serie de objetos para manejar una solicitud. Cada objeto en la cadena contiene lógica que define los tipos de solicitudes que puede manejar; las demás son pasadas al siguiente objeto en la cadena. Es útil cuando se necesita pasar una solicitud a través de una serie de objetos potenciales manejadores hasta que uno de ellos la maneje. Esto puede ser útil en situaciones donde no se sabe de antemano cuál objeto manejará la solicitud, o cuando hay varios objetos que podrían manejar la solicitud (Martínez Canelo, 2020).

El sistema de menús de Drupal sigue el patrón de la cadena de responsabilidad. En cada solicitud de página, el sistema de menús determina si hay un módulo para manejar la solicitud, si el usuario tiene acceso al recurso solicitado y qué función se llamará para hacer el trabajo. Para esto, se pasa un mensaje al elemento del menú correspondiente a la ruta de la solicitud. Si el elemento del menú no puede manejar la solicitud, se pasa por la cadena. Esto continúa hasta que un módulo maneja la solicitud, un módulo niega el acceso al usuario o la cadena se agota (Drupal, 2023).

En el siguiente fragmento de código (**Figura 8**), MenuLinkContent es el elemento del menú y tiene métodos para manejar la solicitud, verificar el acceso del usuario y obtener la URL del recurso solicitado. Si el elemento del menú no puede manejar la solicitud, se pasa por la cadena hasta que otro elemento del menú puede manejarla.

```php
namespace Drupal\Core\Menu;

use Symfony\Component\HttpFoundation\Request;

class MenuLinkContent extends EntityBase {

 public function access(AccountInterface $account, $return_as_object = FALSE)
   $access = AccessResult::neutral();
   $access->addCacheableDependency($this);
   return $access;
 }

 public function getUrlObject() {
   $route_name = $this->getRouteName();
   $route_parameters = $this->getRouteParameters();
   $url = Url::fromRoute($route_name, $route_parameters);
   return $url;
 }

 public function getRouteName() {
   return $this->route_name;
 }

 public function getRouteParameters() {
   return $this->route_parameters;
 }
}
```

Figura 8: Ejemplo de empleo del patrón Chain of responsibility

- **Command (Comando):**

Este patrón encapsula una solicitud como un objeto, permitiendo el parámetro de clientes con diferentes solicitudes y soporte para operaciones deshacer. En este patrón, se crea un objeto de comando que contiene toda la información necesaria para realizar una acción específica, incluyendo el método a llamar, los argumentos a pasar y el receptor que realizará la acción. Su función es desacoplar el objeto que invoca la operación del que sabe cómo realizarla. Esto permite que el objeto invocador se parametrice fácilmente con diferentes comandos y permite la separación de preocupaciones entre el invocador y el receptor del comando (Martínez Canelo, 2020).

En el siguiente ejemplo (**Figura 9**), SendEmailCommand es una clase que representa un comando para enviar un correo electrónico. El método configure() se utiliza para configurar el nombre y la descripción del comando. El método execute() se utiliza para definir la lógica del comando.

```php
namespace Drupal\rename_module\Command;

use Symfony\Component\Console\Command\Command;
use Symfony\Component\Console\Input\InputInterface;
use Symfony\Component\Console\Output\OutputInterface;

class SendEmailCommand extends Command {
 protected function configure() {
 $this
 ->setName('send:email')
 ->setDescription('Send an email');
 }

 protected function execute(InputInterface $input, OutputInterface $output) {
 // Your command logic here
 }
}
```

Figura 9: Ejemplo de empleo del patrón Command

2.7. Tarjetas Clase-Responsabilidad-Colaboración (CRC)

En la fase de Diseño, la metodología XP no requiere la representación del sistema mediante diagramas de clases utilizando la notación de Lenguaje Unificado de Modelado (UML). En su lugar se usan otras técnicas como las tarjetas Clase Responsabilidad Colaboración (CRC) como una extensión informal a UML. La técnica de las tarjetas CRC se puede usar para guiar el sistema a través de análisis guiados por la responsabilidad. Las clases se examinan, se filtran y se refinan en base a sus responsabilidades con respecto al sistema, y las clases con las que necesitan colaborar para completar sus responsabilidades (Wallace, Raggett, & Aufgang, 2003).

El uso de Drupal facilita la creación de nodos que constituyen esencialmente entradas de contenido que pueden ser de cualquier tipo, como un libro, una imagen e incluso un usuario. Los nodos pueden tener campos asociados para almacenar y organizar diferentes tipos de información. Por ejemplo, se puede tener un nodo de tipo "solicitud" con campos para el nombre y el solapín.

Los nodos se pueden crear y gestionar en Drupal a través de la interfaz de administración. Además, se pueden personalizar creando nuevos tipos de contenido y asignándoles diferentes campos.

El uso de taxonomías en Drupal permite clasificar y organizar el contenido. Son esencialmente listas de términos que se pueden utilizar para categorizar nodos. Por

ejemplo, se puede tener una taxonomía llamada "Categoría" con términos como "Estudiante" y "Profesor o especialista". Estos términos pueden tener subcategorías, creando así una estructura jerárquica. El empleo de taxonomías favorece la organización de contenido, la navegación del sitio, y la clasificación de contenido similar.

En resumen, los nodos y las taxonomías en Drupal son herramientas poderosas para gestionar y organizar el contenido de un sitio web. Los nodos son la unidad básica de contenido, mientras que las taxonomías proporcionan una forma de clasificar y organizar este contenido.

Drupal almacena y gestiona la información en varias tablas dentro de la base de datos, cada una de las cuales representa un tipo de entidad o contenido, para esta investigación se emplean los siguientes nodos y taxonomías:

Nodo de libro: Este nodo representará cada libro en el sistema. Se agregaron campos personalizados a este nodo para almacenar detalles del libro, como el título y la cantidad total.

Nodo de usuario: Este nodo representará a cada usuario en el sistema. Se pueden agregar campos personalizados a este nodo para almacenar detalles del usuario, como el nombre y el correo electrónico.

Nodo de solicitud: Este nodo representará cada entrega o devolución de un libro. Se pueden agregar campos personalizados a este nodo para almacenar detalles de la entrega o devolución, como el solapín y la categoría.

Taxonomía de categoría: Esta taxonomía se utiliza para categorizar a los estudiantes y trabajadores que realizan solicitudes en el sistema, por ejemplo, por categoría (Estudiantes o Profesores y especialistas) y por carrera (ICI, ICS, ARSI).

A continuación, se expone un ejemplo de tarjeta CRC obtenida durante la etapa de Diseño del presente trabajo a partir del análisis de los nodos y taxonomías empleados. El resto de las mismas se encuentran en el Anexo 2.

Tabla 11: Tarjeta CRC para la clase Nodo

Tarjeta CRC #1	
Clase: Libro	
Responsabilidad	**Colaboración**

• Guardar y gestionar la información vinculada a los libros. • Mostrar detalles del libro. • Disparar eventos relacionados con la creación y actualización de libros.	▪ Taxonomía: Para acceder a los términos de taxonomía asociados al libro. ▪ Solicitud: Para registrar la bibliografía solicitada.

2.8. Modelo de datos

Un modelo de base de datos es la estructura lógica que adopta la base de datos, incluyendo las relaciones y limitaciones que determinan cómo se almacenan y organizan y cómo se accede a los datos. Así mismo, un modelo de base de datos también define qué tipo de operaciones se pueden realizar con los datos (Ayudaley, 2023).

El lenguaje de consultas determina cómo se manipulan los datos y también proporciona la base para el diseño de la base de datos. En términos generales, casi todos los modelos de base de datos se pueden representar mediante un diagrama de base de datos.

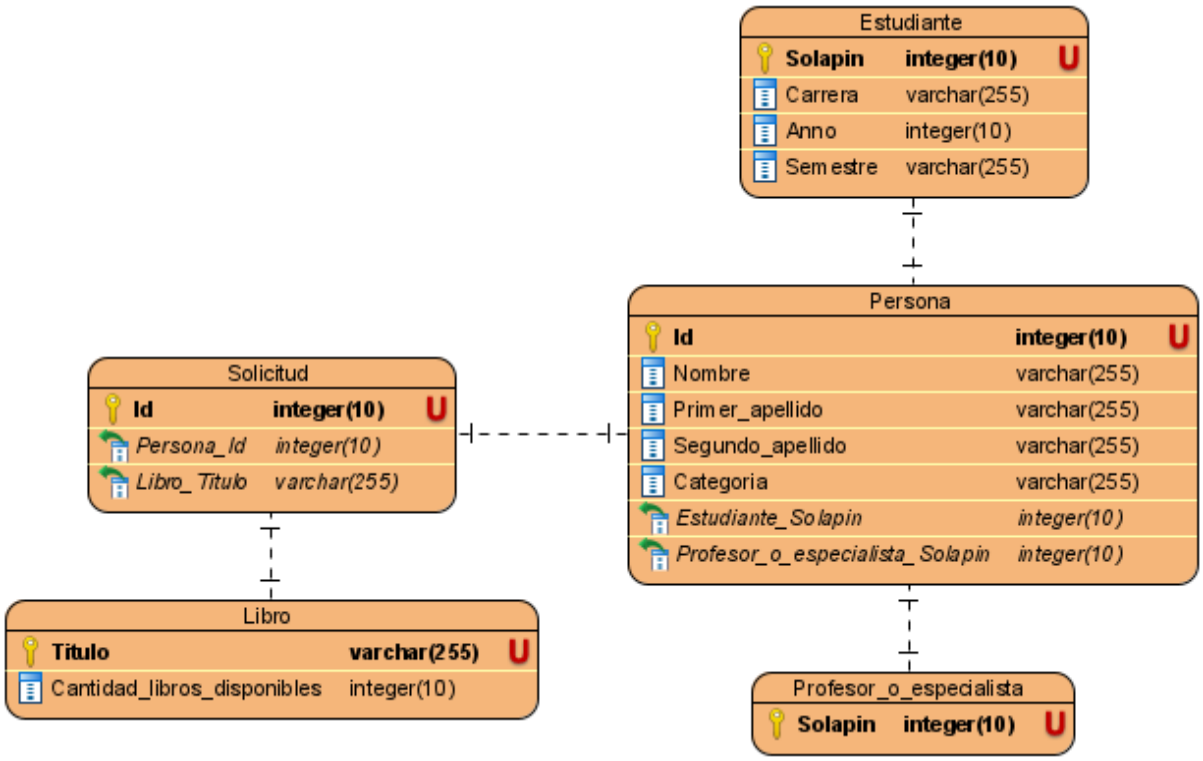


Figura 10: Diagrama del modelo de datos

2.9. Conclusiones del capítulo

En este capítulo se le realizó un análisis de la forma actual de desarrollarse el proceso de entrega y devolución de bibliografía física docente en la Facultad 2 de la UCI. Basándose en este análisis se propuso una solución, que consiste en una aplicación web que automatice el proceso mencionado anteriormente, dándole cumplimiento al segundo objetivo de esta investigación. Se especificó que el sistema debe contar con módulos y roles definidos.

Capítulo 3

Validación de la propuesta de solución

3.1. Introducción

En el presente capítulo se validarán los requisitos y se abordarán las pruebas realizadas al software siguiendo las buenas prácticas de la metodología XP, para garantizar el correcto funcionamiento y la calidad del sistema. Se detallan además las tres iteraciones llevadas a cabo durante la etapa de construcción del sistema, exponiendo las tareas generadas por cada historia de usuario, así como las pruebas unitarias y de aceptación efectuadas sobre el proyecto.

Una vez concluida la fase de pruebas se podrá definir con exactitud, en qué grado de funcionamiento y rendimiento se encuentra el sistema, compararlo con las exigencias del cliente y determinar si se cumplieron sus expectativas.

Fase 4: Implementación y Pruebas

3.2. Estándares de codificación

Los estándares de codificación incorporan principios de ingeniería sólidos para la programación en sus respectivos lenguajes y forman la base de cualquier enfoque preventivo. El costo de un buen software es menor que el costo de un software malo (Hiken, 2023). A continuación, se detallan los estándares de codificación utilizados en la implementación de la solución propuesta.

Identación: Consiste en insertar espacios en blanco o tabuladores en determinadas líneas de código para facilitar su comprensión. En programación se emplea identación para anidar elementos. En Drupal se debe identar con 2 espacios, nunca con tabuladores (Manz, 2021). Además, no se debe dejar en blanco al final de cada línea. En el siguiente ejemplo se presenta un fragmento de código que demuestra la identación del código.

```
*/
function inotu_b5_form_alter(&$form, &$form_state, $form_id) {
  if ($form_id === 'search_block_form') {
    $form['actions']['submit']['#value'] = '';
    $form['actions']['submit']['#attributes']['class'][] = 'fas';
    $form['actions']['submit']['#attributes']['class'][] = 'fa-search';
  }
}
```

Figura 11: Ejemplo de identación de código

Operadores: Al trabajar con cualquier lenguaje de programación, es muy habitual hacer uso de los llamados operadores. Se trata de símbolos que nos permitirán hacer una serie de operaciones con uno o más operadores (generalmente números). Entre ellos se establecen los aritméticos: operadores para realizar operaciones matemáticas, asignación: operadores para guardar información en variables, unarios: operadores que se utilizan con un sólo operando, comparación: operadores para realizar comprobaciones, binarios: operadores a bajo nivel (a nivel de bits) (Manz, 2021). A continuación, se puede apreciar lo antes explicado en un ejemplo del código de la solución.

```
$('.container-mapa').mousemove(function(e) {
  var parentOffset = $(this).offset();
  var x = e.pageX - parentOffset.left;
  var y = e.pageY - parentOffset.top;
  var centerX = $(this).width() / 2;
  var centerY = $(this).height() / 2;
  var deltaX = (centerX - x) / centerX;
  var deltaY = (centerY - y) / centerY;
  $(this).find('img').css({
    'transform': 'scale(1.4) translate(' + deltaX * 110 + 'px,' + deltaY * 110 + 'px)'
  });
});
```

Figura 12: Ejemplo de operadores

Uso de comillas: Las comillas es un recurso usado continuamente cuando estamos desarrollando. Su empleo suele ser algo confuso. Las comillas se usan generalmente cuando se encapsulan o concatenan textos. Existen dos tipos de comillas: las simples (‘cadena’) y las dobles (“cadena”). Y es aquí donde recae la complejidad de su uso, puesto que la incorrecta combinación de estas es la causante de muchos errores (Learning, 2023). A continuación, se evidencia el uso de las comillas en la codificación de la propuesta de solución.

```
26      $('.container-mapa').mouseleave(function() {
27        $(this).find('img').css({
28          'transform': 'scale(1) translate(0, 0)'
29        });
```

Figura 13: Ejemplo de comillas

Uso de punto y coma (;) en código PHP: Aunque PHP permite escribir líneas de código individuales sin el terminador de línea (;), como por ejemplo <?php print $title ?>. En Drupal es siempre obligatorio: <?php print $title; ?> (Drupal, 2023). A continuación, se evidencia el uso de punto y coma en la codificación de la propuesta de solución.

```
30    public function buildForm(array $form, FormStateInterface $form_state) {
31      $config = $this->config('back_to_top.settings');
32      $settings = $config->get();
33
```

Figura 14: Ejemplo de punto y coma

Estructuras de control: Con respecto a las estructuras de control, hay que tener en cuenta las siguientes normas (Drupal, 2023):

- Debe haber un espacio entre el comando que define la estructura (*if, while, for*, etc.) y el paréntesis de apertura. Esto es así para no confundir las estructuras de control con la nomenclatura de las funciones.
- La llave de apertura {se situará en la misma línea que la definición de la estructura, separada por un espacio.
- Se recomienda usar siempre las llaves {} aún en los casos en que no sea obligatorio su uso (una sola "línea" de código dentro de la estructura de control).
- Las estructuras *else* y *elseif* se escribirán en la línea siguiente al cierre de la sentencia anterior.

A continuación, se evidencia el uso de estructuras de control en la codificación de la propuesta de solución.

```
26      if (!Drupal::currentUser()->isAuthenticated() and getConfig('config')) {
27          return new TrustedRedirectResponse($this->getCodeChallenge(), 302);
28      } else {
29          if (Drupal::currentUser()->isAuthenticated()) {
30              return $this->redirect('user.page');
31          } else {
32              return $this->redirect('<front>');
33          }
34      }
```

Figura 15: Ejemplo de estructuras de control

Funciones: Los nombres de las funciones deben estar escritos en minúsculas y las palabras separadas por guion bajo. Además, se debe incluir siempre como prefijo el nombre del módulo o tema, para evitar así duplicidad de funciones. En su declaración, después del nombre de la función, el paréntesis de inicio de los argumentos debe ir sin espacio. Cada argumento debe ir separado por un espacio, después de la coma del argumento anterior (Drupal, 2023). A continuación, se evidencia el uso de funciones en la codificación de la propuesta de solución.

```
24      public function login()
25      {
26          if (!Drupal::currentUser()->isAuthenticated() and getConfig('config')) {
27              return new TrustedRedirectResponse($this->getCodeChallenge(), 302);
28          } else {
29              if (Drupal::currentUser()->isAuthenticated()) {
30                  return $this->redirect('user.page');
31              } else {
32                  return $this->redirect('<front>');
33              }
34          }
35      }
```

Figura 16: Ejemplo de funciones

Comentar el código: Para la realización de comentarios se emplea ***/*/*** para comentarios en varias líneas y ***//*** para comentarios de una única línea. Se deben escribir frases completas, comenzándolas con mayúscula y terminándolas con un punto. En caso de que en el comentario se haga referencia a una constante, esta deberá escribirse en mayúscula (por ejemplo: TRUE o FALSE) (Drupal, 2023).

```
36  /**
37   * Implements hook_form_alter().
38   */
39  function inotu_b5_form_alter(&$form, &$form_state, $form_id) {
40    if ($form_id === 'search_block_form') {
41      // Utilizamos el icono "search" de Font Awesome como fondo del botón de envío.
42      $form['actions']['submit']['#value'] = '';
```

Figura 17: Ejemplo de comentarios en el código

3.3. Desarrollo por iteraciones

Durante el transcurso de las iteraciones se realiza la implementación de las historias de usuario seleccionadas para cada una de estas. Al inicio de las mismas, se lleva a cabo una revisión del plan de iteraciones y se modifica de ser necesario. Como parte de este plan, se descomponen las HU en Tareas de Ingeniería. Estas tareas son para el uso de los programadores, pueden escribirse utilizando un lenguaje técnico y no necesariamente deben ser entendibles para el cliente (Beck, 2000).

Se llevaron a cabo una serie de iteraciones de desarrollo sobre el sistema según la planificación realizada, obteniéndose al finalizar un producto listo para su despliegue. A continuación, se detallan cada una de las iteraciones.

3.4. Iteración 1

Esta iteración tiene como objetivo la implementación de las historias de usuario de prioridad media. Durante el transcurso de la misma se creará la base de la arquitectura del sistema con una funcionalidad mínima. Al final de esta se contará con una primera versión de prueba, la cual será mostrada al cliente con el objetivo de obtener una retroalimentación para el equipo de desarrollo.

3.4.1. Tareas de ingeniería

3.4.1.1. Historia de usuario “Autenticar usuario”

Tabla 12: Tarea de Ingeniería "API de autenticación"

Tarea de Ingeniería	
Nombre: Interfaz de autenticación	
HU: Autenticar usuario	**Puntos estimados:** 0.2
Fecha de inicio: 7/08/2023	**Fecha de fin:** 8/08/2023
Descripción: Se debe proveer una interfaz de autenticación, donde las credenciales a recibir son el nombre de usuario y contraseña. Los datos del usuario autenticado deben guardarse en la base de datos local para permitir el acceso sin conexión.	

Tabla 13: Tarea de Ingeniería "Autenticar usuario"

Tarea de Ingeniería	
Nombre: Autenticar usuario	
HU: Autenticar usuario	**Puntos estimados:** 0.6
Fecha de inicio: 8/08/2023	**Fecha de fin:** 10/08/2023
Descripción: Se debe mostrar un formulario de autenticación para que el usuario inserte sus credenciales (usuario y contraseña). Los datos deberán ser válidos para autenticarse, mostrando un mensaje de error en caso contrario. No se permitirá el	

acceso a las funcionalidades del sistema hasta que el usuario no se haya autenticado.

Tabla 14: Tarea de Ingeniería "Cerrar sesión"

Tarea de Ingeniería	
Nombre: Cerrar sesión	
HU: Autenticar usuario	**Puntos estimados:** 0.2
Fecha de inicio: 10/08/2023	**Fecha de fin:** 11/08/2023
Descripción: Se debe permitir al usuario cerrar la sesión desde cualquier página, redirigiéndolo inmediatamente hacia la página de acceso.	

3.4.1.2. Historia de usuario “Gestionar usuarios”

Tabla 15: Tarea de Ingeniería "API para los usuarios"

Tarea de Ingeniería	
Nombre: Usuarios	
HU: Gestionar usuarios	**Puntos estimados:** 0.2
Fecha de inicio: 14/08/2023	**Fecha de fin:** 15/08/2023
Descripción: Se debe proveer una interfaz para gestionar los usuarios creados por el administrador, garantizando un correcto funcionamiento de los procesos de acceso a la información.	

Tabla 16: Tarea de Ingeniería "Añadir usuario"

Tarea de Ingeniería	
Nombre: Añadir usuario	
HU: Gestionar usuarios	**Puntos estimados:** 0.6
Fecha de inicio: 15/08/2023	**Fecha de fin:** 17/08/2023
Descripción:	

El administrador puede añadir un usuario llenando el formulario correspondiente con los datos requeridos (dirección de correo electrónico, nombre, contraseña, rol).

Tabla 17: Tarea de Ingeniería "Modificar usuario"

Tarea de Ingeniería	
Nombre: Editar usuario	
HU: Gestionar usuarios	**Puntos estimados:** 0.6
Fecha de inicio: 17/08/2023	**Fecha de fin:** 21/08/2023
Descripción: El administrador puede editar un usuario del listado, permitiendo cambiar los datos del mismo y actualizar el contenido asociado.	

Tabla 18: Tarea de Ingeniería "Eliminar usuario"

Tarea de Ingeniería	
Nombre: Eliminar usuario	
HU: Gestionar usuarios	**Puntos estimados:** 0.4
Fecha de inicio: 22/08/2023	**Fecha de fin:** 24/08/2023
Descripción: El administrador puede eliminar un usuario del listado. Esta acción debe confirmarse debido a que es irreversible.	

Tabla 19: Tarea de Ingeniería "Listar usuarios"

Tarea de Ingeniería	
Nombre: Listar usuario	
HU: Gestionar usuarios	**Puntos estimados:** 0.2
Fecha de inicio: 24/08/2023	**Fecha de fin:** 25/08/2023
Descripción: El administrador puede ver una tabla con los usuarios que están registrados, permitiendo gestionar (editar, mostrar, eliminar) cada uno de ellos.	

3.5. Iteración 2

El objetivo de esta iteración es la implementación de las historias de usuario de prioridad alta. Al finalizar la misma se contará con una versión funcional del sistema, donde se podrán probar las funcionalidades relacionadas con la entrega y devolución de libros. Esta versión igualmente se mostrará al cliente con el objetivo de realizar cambios necesarios en base a la opinión del mismo.

3.5.1. Tareas de Ingeniería

3.5.1.1. Historia de usuario “Gestionar libros”

Tabla 20: Tarea de Ingeniería "API para los libros"

Tarea de Ingeniería	
Nombre: Listado de libros	
HU: Gestionar libros	**Puntos estimados:** 0.2
Fecha de inicio: 28/08/2023	**Fecha de fin:** 29/08/2023
Descripción: Se debe proveer una interfaz para gestionar los libros creados por el usuario, garantizando un correcto funcionamiento de los procesos de acceso a la información.	

Tabla 21: Tarea de Ingeniería "Añadir libro"

Tarea de Ingeniería	
Nombre: Añadir libro	
HU: Gestionar libros	**Puntos estimados:** 0.6
Fecha de inicio: 29/08/2023	**Fecha de fin:** 31/08/2023
Descripción: Un usuario puede añadir un libro llenando el formulario correspondiente con los datos requeridos (nombre, cantidad).	

Tabla 22: Tarea de Ingeniería "Modificar libro"

Tarea de Ingeniería

Nombre: Modificar libro	
HU: Gestionar libros	**Puntos estimados:** 0.6
Fecha de inicio: 31/08/2023	**Fecha de fin:** 4/09/2023
Descripción: Un usuario puede editar un libro del listado, permitiendo cambiar los datos del mismo y actualizar el contenido asociado.	

Tabla 23: Tarea de Ingeniería "Eliminar libro"

Tarea de Ingeniería	
Nombre: Eliminar libro	
HU: Gestionar libros	**Puntos estimados:** 0.4
Fecha de inicio: 5/09/2023	**Fecha de fin:** 7/09/2023
Descripción: Un usuario puede eliminar un libro del listado. Esta acción debe confirmarse debido a que es irreversible.	

Tabla 24: Tarea de Ingeniería "Listar libros"

Tarea de Ingeniería	
Nombre: Listar libros	
HU: Gestionar libros	**Puntos estimados:** 0.2
Fecha de inicio: 7/09/2023	**Fecha de fin:** 8/09/2023
Descripción: Un usuario puede ver una tabla con los libros que están registrados, permitiendo gestionar (editar, mostrar, eliminar) cada una de ellos.	

3.5.1.2. Historia de usuario “Gestionar solicitudes”

Tabla 25: Tarea de Ingeniería "API para las solicitudes"

Tarea de Ingeniería
Nombre: Interfaz para las solicitudes

HU: Gestionar solicitudes	**Puntos estimados:** 0.2
Fecha de inicio: 11/09/2023	**Fecha de fin:** 12/09/2023
Descripción: Se debe proveer una interfaz para gestionar las solicitudes creadas por el usuario, garantizando un correcto funcionamiento de los procesos de acceso a la información.	

Tabla 26: Tarea de Ingeniería "Añadir solicitud"

Tarea de Ingeniería	
Nombre: Añadir solicitud	
HU: Gestionar solicitudes	**Puntos estimados:** 0.6
Fecha de inicio: 12/09/2023	**Fecha de fin:** 14/09/2023
Descripción: Un usuario puede añadir una solicitud llenando el formulario correspondiente con los datos requeridos (nombre, primer apellido, segundo apellido, categoría, solapín, carrera (en caso de ser estudiante), nombre del libro, año y semestre).	

Tabla 27: Tarea de Ingeniería "Modificar solicitud"

Tarea de Ingeniería	
Nombre: Modificar solicitud	
HU: Gestionar solicitudes	**Puntos estimados:** 0.6
Fecha de inicio: 14/09/2023	**Fecha de fin:** 18/09/2023
Descripción: Un usuario puede editar una solicitud del listado, permitiendo cambiar los datos de la misma y actualizar el contenido asociado.	

Tabla 28: Tarea de Ingeniería "Eliminar solicitud"

Tarea de Ingeniería	
Nombre: Eliminar solicitud	
HU: Gestionar solicitudes	**Puntos estimados:** 0.2

Fecha de inicio: 19/09/2023	**Fecha de fin:** 20/09/2023
Descripción: Un usuario puede eliminar una solicitud del listado. Esta acción debe confirmarse debido a que es irreversible.	

Tabla 29: Tarea de Ingeniería "Listar solicitudes"

Tarea de Ingeniería	
Nombre: Listar solicitudes	
HU: Gestionar solicitudes	**Puntos estimados:** 0.2
Fecha de inicio: 20/09/2023	**Fecha de fin:** 21/09/2023
Descripción: Un usuario puede ver una tabla con las solicitudes que están registradas, permitiendo gestionar (editar, mostrar, eliminar) cada una de ellas.	

Tabla 30: Tarea de Ingeniería "Mostrar solicitudes"

Tarea de Ingeniería	
Nombre: Mostrar solicitudes	
HU: Gestionar solicitudes	**Puntos estimados:** 0.2
Fecha de inicio: 21/09/2023	**Fecha de fin:** 22/09/2023
Descripción: Un usuario puede ver el contenido asociado a una solicitud del listado.	

3.6 Iteración 3

Durante el transcurso de esta iteración se implementa la historia de usuario de prioridad media. Al finalizar la misma se contará con la primera versión del producto final, adicionando a las funcionalidades anteriores todo lo concerniente al sistema de gestión de bibliografía física docente. Como resultado de esta, el sistema estará listo para su despliegue.

3.6.1. Tareas de Ingeniería

3.6.1.1. Historia de usuario “Generar inventario”

Tarea de Ingeniería	
Nombre: Interfaz para el Inventario	
HU: Generar inventario	**Puntos estimados:** 0.6
Fecha de inicio: 25/09/2023	**Fecha de fin:** 27/09/2023
Descripción: Se debe proveer una interfaz que muestre el inventario total de solicitudes, incluyendo la cantidad de libros existentes en el almacén y la cantidad de libros entregados, garantizando un correcto funcionamiento de los procesos de acceso a la información.	

Tarea de Ingeniería	
Nombre: Imprimir	
HU: Generar inventario	**Puntos estimados:** 0.4
Fecha de inicio: 28/09/2023	**Fecha de fin:** 29/09/2023
Descripción: Un usuario puede obtener un reporte en formato digital (PDF) y físico (documento impreso) del inventario total de solicitudes.	

3.7. Pruebas

A las pruebas se le confiere un valor esencial dentro del desarrollo de aplicaciones ya que un fallo en estas puede representar costos elevados. Las pruebas son una actividad en la cual un sistema o componente es ejecutado bajo unas condiciones o requerimientos especificados, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema o componente. Para la validación de un software de manera general se utilizan principalmente dos métodos de prueba: caja blanca y caja negra. Las pruebas de caja blanca basan su funcionamiento en evaluar la información sobre cómo el software ha sido diseñado y codificado. El método caja negra actúa sobre la validación de los requisitos funcionales y se aplican a la interfaz del software para examinar algún aspecto funcional de un sistema, por lo que también se les denominan pruebas funcionales o de aceptación (Bertolino, 2007).

3.7.1. Pruebas de Caja Negra

Las pruebas de Caja Negra deben su nombre a los elementos que estas revisan y las condiciones en que se hace la revisión. Estas se basan en los requerimientos funcionales del sistema y se llevan a cabo desde el exterior de la aplicación. Este tipo de prueba es importante a la hora de medir el grado de cumplimiento de los requerimientos solicitados por el cliente y se aplican sobre la interfaz de la aplicación, observando las respuestas del sistema ante determinadas acciones (PRESSMAN, 2001).

3.7.1.1. Pruebas de aceptación

Las pruebas de aceptación son un nivel de pruebas que emplean en mayor medida técnicas de caja negra, las cuales son definidas por el cliente para cada historia de usuario, estas se realizan antes finalizar cada iteración. El cliente es el encargado de realizar dichas pruebas verificando con distintas entradas de datos, que el sistema responda a estas adecuadamente, y que cumpla con las funcionalidades deseadas. Para la correcta realización de estas pruebas, se empleó la **técnica de Partición Equivalente**.

La técnica de Partición Equivalente es una técnica de prueba que se utiliza para reducir el número de casos de prueba necesarios para probar un sistema. Esta técnica se basa en dividir el conjunto de datos de entrada en grupos o particiones que se espera que tengan un comportamiento similar. Luego, se selecciona un caso de prueba de cada partición para probar el sistema (PRESSMAN, 2001).

El objetivo de esta técnica es garantizar que se prueben todas las condiciones posibles dentro de cada partición, sin necesidad de probar cada combinación de datos de entrada. Esto ayuda a reducir el tiempo y los recursos necesarios para realizar las pruebas (PRESSMAN, 2001).

Para aplicar la técnica de Partición Equivalente, se deben seguir los siguientes pasos:

1. Identificar los diferentes conjuntos de datos de entrada que se pueden utilizar en el sistema. Por ejemplo, si el sistema acepta números enteros, se pueden identificar las particiones de números positivos, negativos y cero.
2. Definir las condiciones límite para cada partición. Estas condiciones límite son los valores extremos que se pueden utilizar en cada partición. Por ejemplo, para la partición de números positivos, se pueden definir las condiciones límite como el número más pequeño y el número más grande que se pueden aceptar.

3. Seleccionar un caso de prueba de cada partición. Este caso de prueba debe representar el comportamiento típico de la partición. Por ejemplo, para la partición de números positivos, se puede seleccionar el número 10 como caso de prueba.
4. Ejecutar los casos de prueba seleccionados y verificar si el sistema se comporta como se espera en cada partición.

Tabla 31: Clases de equivalencia para la TI "Añadir solicitud"

Identificador	Entrada (clase)	Clases válidas	Clases inválidas
Nombre	Nombre	Nombre = a-z	Nombre = vacío
Primer apellido	Apellido	Apellido = a-z	Apellido = vacío
Segundo apellido	Apellido	Apellido = a-z	Apellido = vacío
Solapín	Solapín	Solapín = 0-9	Solapín = vacío
Categoría	Categoría	Categoría = seleccionado	Categoría = sin selección
Nombre del libro	Título	Título = seleccionado	Título = sin selección
Año	Año	Año = seleccionado	Año = sin selección
Semestre	Semestre	Semestre = seleccionado	Semestre = sin selección

Tabla 32: Caso de prueba para la TI " Añadir solicitud"

<table>
<tr><td>Escenario: Operación de Añadir solicitud.</td><td rowspan="2">Descripción: Selecciona la opción Añadir solicitud.</td><td>Respuesta del sistema</td><td>Flujo central</td></tr>
<tr><td>Identificadores</td><td rowspan="9">Brinda la posibilidad de introducir o seleccionar de manera obligatoria los siguientes datos:
Nombre
Primer apellido
Segundo apellido
Solapín
Categoría
Nombre del libro
Año</td><td rowspan="9">/node/add/entregar_libro</td></tr>
<tr><td>Nombre</td><td>-</td></tr>
<tr><td>Primer apellido</td><td>-</td></tr>
<tr><td>Segundo apellido</td><td>-</td></tr>
<tr><td>Solapín</td><td>-</td></tr>
<tr><td>Categoría</td><td>-</td></tr>
<tr><td>Nombre del libro</td><td>-</td></tr>
<tr><td>Año</td><td>-</td></tr>
<tr><td>Semestre</td><td>-</td></tr>
</table>

		• Semestre	
Escenario: Opción de Guardar. **Identificadores**	**Descripción:** Introduce y/o selecciona los datos de una solicitud y selecciona la opción Guardar.	**Respuesta del sistema**	**Flujo central**
Nombre	Yadian	Valida los datos, crea una nueva solicitud, muestra una vista previa de la solicitud creada y muestra un mensaje de información (**Mensaje de estado:** Solicitud *Yadian Tresgallos González* ha sido creada.	/node/add/entregar_libro
Primer apellido	Tresgallos		
Segundo apellido	González		
Solapín	E-18480		
Categoría	Estudiante		
Nombre del libro	Matemática Discreta		
Año	5		
Semestre	Décimo		
Escenario: Opción de Guardar con datos incompletos. **Identificadores**	**Descripción:** Introduce y/o selecciona alguno de los datos de una solicitud y selecciona la opción Guardar.	**Respuesta del sistema**	**Flujo central**
Nombre	Yadian	Muestra un indicador sobre los campos vacíos y muestra un mensaje de información (**Mensaje de error:** El campo es obligatorio).	/node/add/entregar_libro
Primer apellido	Tresgallos		
Segundo apellido	-		
Solapín	E-		
Categoría	Estudiante		
Nombre del libro	Matemática Discreta		
Año	5		
Semestre	-		
Escenario: Opción de Regresar al sitio. **Identificadores**	**Descripción:** Selecciona la opción de Regresar al sitio.	**Respuesta del sistema**	**Flujo central**
Nombre	Yadian		/node/add/entregar_libro
Primer apellido	Tresgallos		

Segundo apellido	González	Elimina los datos creados y regresa a la página anterior.	
Solapín	E-18480		
Categoría	Estudiante		
Nombre del libro	Matemática Discreta		
Año	5		
Semestre	Décimo		

Es importante tener en cuenta que la técnica de Partición Equivalente no garantiza la detección de todos los errores en el sistema, pero ayuda a reducir el número de casos de prueba necesarios para cubrir todas las condiciones posibles. Las restantes pruebas de aceptación, se encuentran en el Anexo 6.

3.7.2. Pruebas de Caja Blanca

Es una técnica de prueba de software que se centra en examinar y evaluar la estructura interna del código fuente de un programa. En otras palabras, se trata de analizar el funcionamiento interno del software y diseñar pruebas en base a ese conocimiento.

En las pruebas de caja blanca, el tester tiene acceso al código fuente y puede examinar cómo se implementan las funciones, las estructuras de control, las condiciones y las rutas de ejecución. Esto permite diseñar casos de prueba específicos para cubrir diferentes caminos de ejecución, condiciones lógicas y flujos de datos (Bertolino, 2007).

El objetivo principal de las pruebas de caja blanca es garantizar una cobertura exhaustiva del código, es decir, asegurarse de que todas las instrucciones, condiciones y caminos de ejecución sean probados. Esto ayuda a identificar posibles errores, fallas o comportamientos inesperados en el software (Bertolino, 2007).

3.7.2.1 Pruebas unitarias

Las pruebas unitarias son la verificación del código y son ejecutadas por los desarrolladores, pues son los encargados de revisar el código directamente. El objetivo de estas pruebas es obtener un código limpio y conciso, que cumpla con cada una de las funcionalidades a las que tributan (Squirrels, 2023).

Las pruebas unitarias son un tipo de prueba de software que se centra en probar unidades individuales de código, como funciones, métodos o clases, de forma aislada. El objetivo principal de las pruebas unitarias es verificar que cada unidad de código

funcione correctamente de manera independiente antes de integrarla con otras partes del sistema.

Se realizan generalmente por los propios desarrolladores y se ejecutan de manera automatizada. Estas pruebas se diseñan para validar el comportamiento esperado de una unidad de código y detectar posibles errores o fallas en su implementación.

Se empleó la **técnica de Camino Básico** como método de prueba, la cual consistió en diseñar casos de prueba para evaluar la complejidad lógica del diseño procedural. Estos casos de prueba se enfocaron en identificar y definir un conjunto fundamental de caminos de ejecución.

Para aplicar la técnica, es necesario utilizar una notación que permita representar el flujo de control. Una forma común de representarlo es mediante un grafo de flujo, en el cual cada nodo representa una acción o decisión, y las flechas indican el flujo de ejecución entre ellos (Bertolino, 2007):

- Cada vértice del grafo representa una o más líneas de código fuente.
- Todo segmento de código de cualquier programa se puede traducir a un grafo de flujo.
- La complejidad ciclomática del grafo se determina mediante un cálculo.

Un grafo de flujo se compone de tres elementos esenciales que facilitan su creación y comprensión (Bertolino, 2007):

- Nodos: Son los círculos representados en el grafo de flujo, el cual representa una o más secuencias del procedimiento, donde un nodo corresponde a una secuencia de procesos o a una sentencia de decisión. Los nodos que no están asociados se utilizan al inicio y final del grafo.
- Aristas: Son constituidas por las flechas del grafo, son iguales a las representadas en un diagrama de flujo y constituyen el flujo de control del procedimiento. Las aristas terminan en un nodo, aun cuando el nodo no representa la sentencia de un procedimiento.
- Regiones: Son las áreas delimitadas por las aristas y nodos donde se incluye el área exterior del grafo, como una región más. Las regiones se enumeran siendo la cantidad de regiones equivalente a la cantidad de caminos independientes del conjunto básico de un procedimiento.

Para realizar la técnica planteada, es necesario calcular antes la complejidad ciclomática del algoritmo o fragmento de código a analizar. Se utiliza la fórmula de McCabe. Se basa

en el número de caminos independientes que existen en el código, se utiliza para medir la complejidad estructural del mismo y se calcula de la siguiente manera:

M = E - N + 2

Donde:

- M es la complejidad ciclomática.
- E es el número de aristas del grafo de flujo del programa.
- N es el número de nodos del grafo de flujo del programa.

En los fragmentos de código que se muestran a continuacion:

- Se declara lo que se conoce como literal o hasches en Twig. Los hashes se definen como una lista de keys y values separados por dos puntos (ejemplo ":") y envueltos por llaves (ejemplo {}).
- Luego se recorre una variable rows para cada fila, la cual extrae la categoría del libro e inicializa una lista vacía de libros.
- A continuación, se recorren los libros en la fila, extrayendo el nombre y la cantidad total de cada libro, y se agregan a la lista de libros.
- Después se verifica si la categoría del libro ya está en el diccionario categoriasLibros.
- Si no está, agrega la categoría y la lista de libros al diccionario.

- Si sí está, combina la lista de libros existente con la nueva lista.

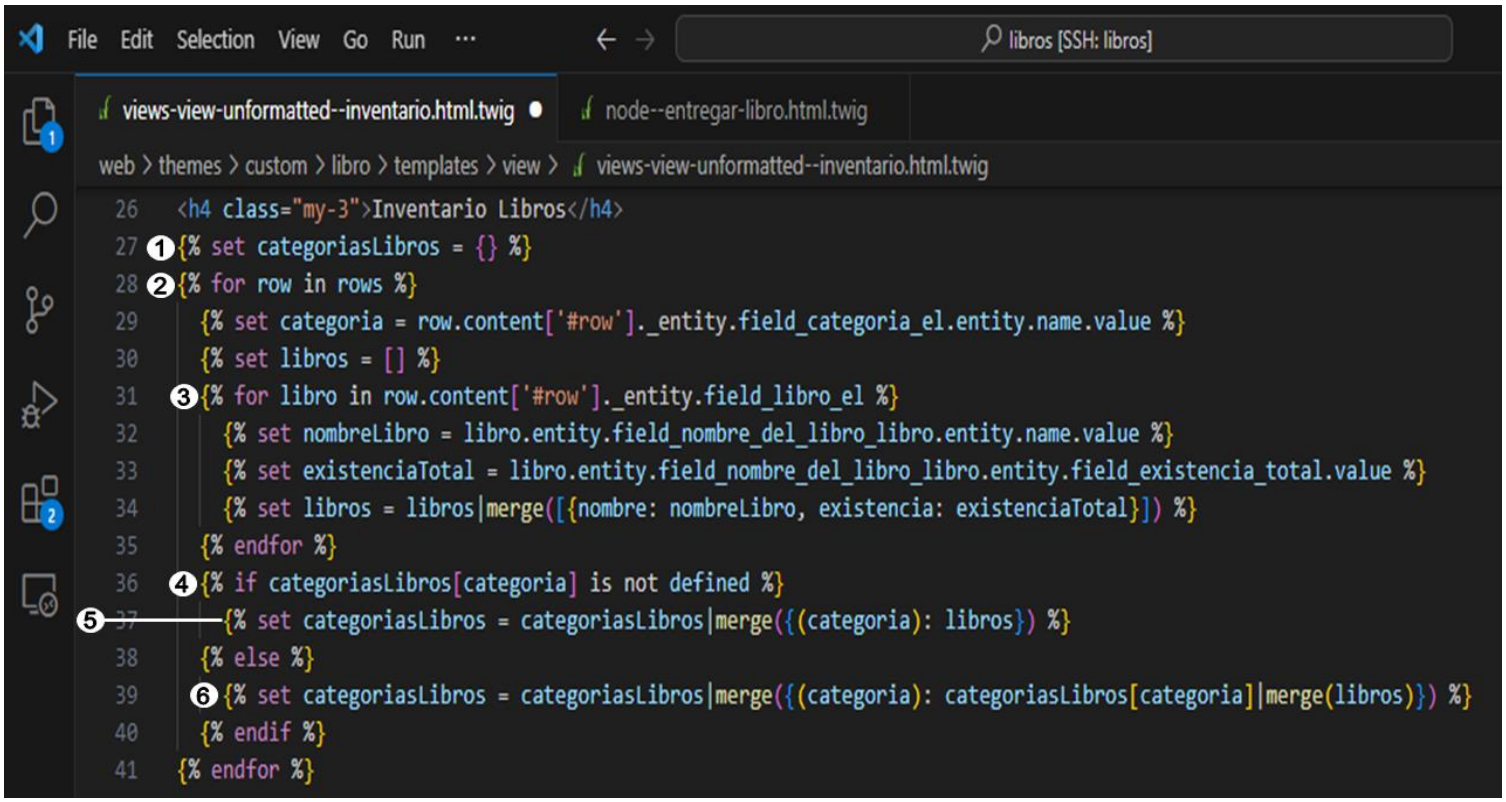


Figura 18: Fragmento de código de la funcionalidad "Generar inventario"

- Se crea una tabla con encabezados para el nombre del libro, categoría, cantidad entregada, total entregado, cantidad en almacén y existencia total.
- Luego se inicializan dos diccionarios vacíos librosUnicos y librosTotales.
- El código luego recorre las categorías y los libros en el diccionario categoriasLibros.
- Para cada libro, verifica si el libro ya está en el diccionario librosUnicos.
- Si no está, agrega el libro y la categoría.
- Si sí está, incrementa el recuento del libro en la categoría.

Figura 19: Continuación de fragmento de código de la funcionalidad "Generar inventario"

El grafo de flujo del programa es una representación gráfica del código, en la que los nodos representan las instrucciones y las aristas representan las posibles rutas de ejecución. Para calcular la complejidad ciclomática, primero se debe construir el grafo de flujo del programa y contar el número de aristas, nodos y componentes conectados. Luego, se aplica la fórmula de McCabe para obtener el valor de M.

La construcción del grafo de flujo se estructuró a partir del fragmento de código perteneciente a la funcionalidad "Generar inventario" ilustrada en la Figura 12 y la Figura 13.

M = E - N + 2

M = (9 - 7) + 2

M = 2 + 2 = **4**

Basado en los resultados obtenidos, se concluye que la funcionalidad tiene una complejidad ciclomática de 4. Esto implica que hay, como máximo 4 rutas lógicas de ejecución, lo que significa que presenta un riesgo bajo para el sistema. A continuación, se muestran.

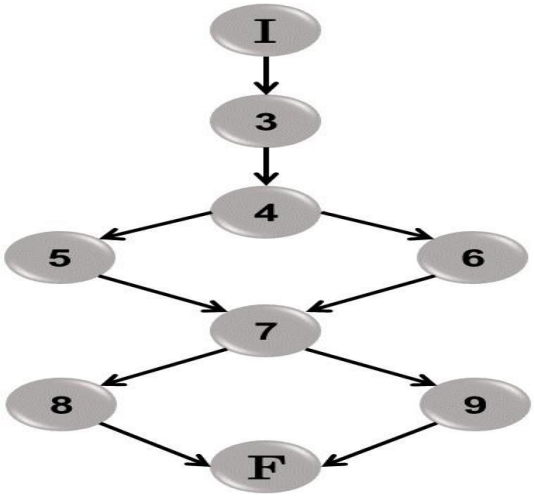


Figura 20: Grafo de flujo

Tabla 33: Técnica del camino básico

No.	Camino básico
1	I-3-4-5-7-8-F
2	I-3-4-5-7-9-F
3	I-3-4-6-7-8-F
4	I-3-4-6-7-9-F

A continuación, se muestran los diseños de casos de prueba que corresponden a los caminos básicos generados durante las pruebas de caja blanca.

Tabla 34: Técnica del camino básico caso de prueba No. 1

Caso de prueba	
Camino	I-3-4-5-7-8-F
Descripción	Mostrar la información asociada al estado de solicitud de un libro (existencia total, cantidad en almacén, total entregado, cantidad entregada).
Condiciones de ejecución	Añadir la solicitud de un libro, presente en otras solicitudes. Tiene que existir al menos una solicitud con la categoría seleccionada (profesor o especialista y/o estudiante).

Entrada	Se introducen todos los datos válidos, correspondientes a la solicitud de un libro.
Resultados esperados	Modifica la cantidad de entregados en la categoría seleccionada.

Tabla 35: Técnica del camino básico caso de prueba No. 2

Caso de prueba	
Camino	I-3-4-5-7-9-F
Descripción	Mostrar la información asociada al estado de solicitud de un libro (existencia total, cantidad en almacén, total entregado, cantidad entregada).
Condiciones de ejecución	No exista ninguna solicitud con la categoría seleccionada (profesor o especialista y/o estudiante).
Entrada	Se introducen todos los datos válidos, correspondientes a la solicitud de un libro.
Resultados esperados	Crea la categoría seleccionada y muestra la cantidad de entregados.

3.8. Resultados de las pruebas

Durante la ejecución de las pruebas, se llevó a cabo una primera iteración en la que se evaluaron las historias de usuario con el objetivo de determinar en qué medida la aplicación cumplía con las funcionalidades implementadas. En esta primera iteración se encontraron un total de 10 No Conformidades (NC), clasificadas en 3 de Ortografía, 5 de Validación y 2 Funcionales. Después de corregir estas NC, se realizó una segunda iteración en la que se identificaron 6 nuevas NC, 2 de Ortografía, 3 de Validación y 1 Funcionales.

Posteriormente, se llevó a cabo una tercera iteración en la que se encontraron 4 nuevas NC, 1 de Ortografía, 2 de Validación y 1 Funcionales. Después de corregir estas deficiencias, se realizó una cuarta iteración de pruebas que arrojó resultados satisfactorios. En base a esto, se decidió no realizar más iteraciones de pruebas y se consideraron concluidas.

Los resultados de las pruebas de aceptación para cada iteración se pueden observar a continuación en un gráfico:

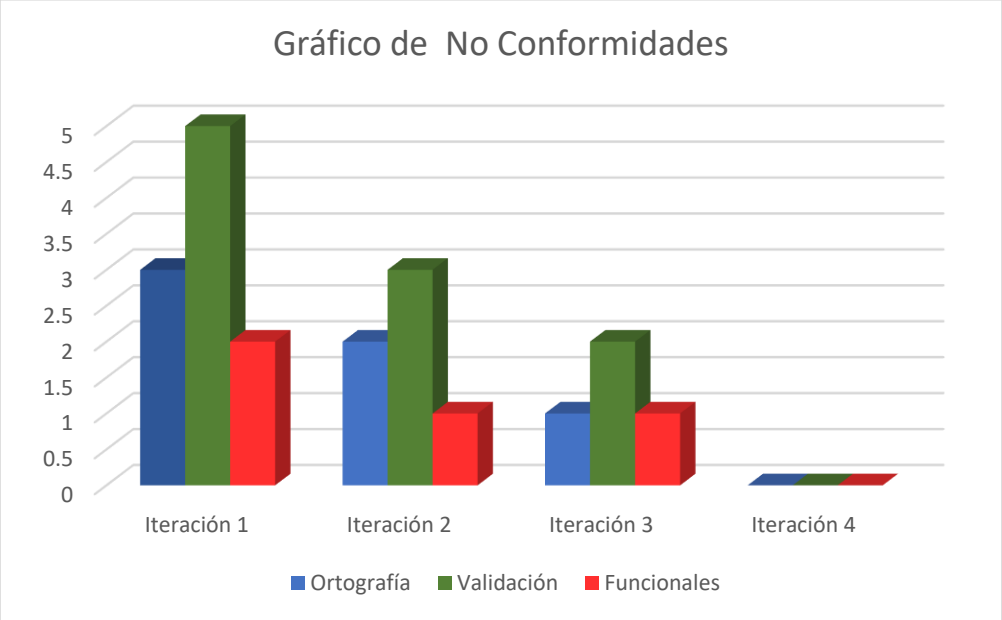


Figura 21: Gráfico de no conformidades

Se identificaron principalmente No Conformidades (NC), como errores ortográficos, omisiones de tildes y cambios de mayúsculas a minúsculas. Además, se encontraron NC relacionadas con errores de validación y validaciones incorrectas en los campos de los formularios. Después de cada iteración, se resolvieron las NC identificadas y se cerraron.

3.9. Conclusiones del capítulo

Después de definir y llevar a cabo una estrategia de pruebas, se llegó a las siguientes conclusiones:

- Al realizar pruebas de software en diferentes niveles, se pudo verificar la correcta implementación del sistema de gestión de libros.
- Al ejecutar pruebas de caja negra utilizando el método de partición de equivalencia y diseñando casos de prueba correspondientes, se logró identificar y eliminar no conformidades que afectaban moderadamente el funcionamiento de la solución de software construida.
- Los resultados de estas pruebas en la plataforma demostraron la efectividad de la solución para su implementación.

Conclusiones generales

La implantación del sistema informático para la gestión de la bibliografía física docente desarrollado en esta investigación, permitirá la fácil consulta y gestión de la información generada a partir de las solicitudes de textos realizadas por estudiantes y trabajadores de la facultad.

Implementar servicios informáticos que mejoren la calidad de cada uno de los procesos que se viven a diario constituye un reto de la rama de la informática.

De esta forma, la información que hoy solo se accede físicamente, será utilizada con el empleo de las nuevas tecnologías de la información y las comunicaciones (TIC), constituyendo un pilar clave en el desarrollo de las mismas y dando de esta manera otro paso de avance a la informatización de la sociedad por la que aboga el país.

Recomendaciones

Para futuras investigaciones, se sugiere lo siguiente:

- Implementar las funcionalidades necesarias para manejar los procesos de pago por pérdida de libros.

Referencias bibliográficas

Acibeiro, M. (30 de 5 de 2023). *Qué es un gestor de contenidos y cuál es el mejor para crear una web*. Recuperado el 19 de 9 de 2023, de https://www.lucushost.com/blog/gestor-de-contenidos/#:~:text=Un%20gestor%20de%20contenidos%20o%20CMS%20(del%20ingl%C3%A9s%2C%20Content%20Management,de%20tener%20conocimientos%20de%20programaci%C3%B3n.

Almarales Lara, B., & Sencial Terrero, R. (2015). *Desarrollo de un portal web para la gestión de servicios en el Consejo Nacional de Patrimonio Cultural.*

Amaya, L. A. (2017). *FUNDAMENTOS DE LENGUAJE UNIFICADO DE MODELADO UML*. Recuperado el 26 de junio de 2023, de https://docplayer.es/94152788-Fundamentos-de-lenguaje-unificado-de-modelado-uml-ing-luis-armando-amaya-q-laboratorio-1-introduccion-a-la-modelacion-del-sistema-en-uml.html

Apache. (6 de 4 de 2023). *Welcome! - The Apache HTTP Server Project*. Recuperado el 18 de 9 de 2023, de https://httpd.apache.org/

Ayudaley. (2023). *El Modelo base de datos: Definición y tipos*. Recuperado el 12 de 9 de 2023, de https://ayudaleyprotecciondatos.es/bases-de-datos/modelos/

Barcia, D. (2003). *Qué es CSS?* Recuperado el 17 de junio de 2023, de http://www.maestrosdelweb.com/introcss/

Beck, K. (2000). *Extreme Programming Explained. Embrace Change.* (Addison Wesley ed.). Pearson Education.

Bertolino, A. (2007). *Software testing research: Achievements, challenges, dreams.*

Bootello, R. (2007-2008). *Sistemas de Información.*

Bootstrap. (12 de 4 de 2020). *Bootstrap: guía para principiantes de qué es, por qué y cómo usarlo*. Recuperado el 18 de 9 de 2023, de https://rockcontent.com/es/blog/bootstrap/

Calero Solís, M. (2003). *Una explicación de la programación extrema (XP).* Madrid: V Encuentro usuarios xBase 2003.

Canós, J. H., Letelier, P., & Carmen Penadés, M. (2011). *Métodologías Ágiles en el Desarrollo de Software.* Valencia: DSIC-Universidad Politécnica de Valencia.

Cañedo Iglesias, C. M., Zamora Fonseca, R., Linares Armas, D., Martínez Santos, K., & Nuñez Chaviano, K. (marzo de 2010). *La biblioteca virtual de la Universidad de Cienfuegos, en Contribuciones a las Ciencias Sociales*. Recuperado el 15 de 8 de 2023, de https://www.eumed.net/rev/cccss/07/ifasc.htm

Capterra. (2019). *CodeAchi Library Management System*. Recuperado el 22 de junio de 2023, de https://www.capterra.com/p/183633/CodeAchi-Library-Management-System/

Capterra. (2020). *ResourceMate*. Recuperado el 22 de junio de 2023, de https://www.capterra.es/software/40056/resourcemate

Carrizo, D., & Ortiz, C. (2016). Modelos del proceso de educción de requisitos: Un mapeo sistemático. *Ingeniería y desarrollo*, 184 - 203.

Chiluisa Pallo, A. P., & Loarte Cajamarca, B. G. (2014). *Desarrollo e implantación del sistema de control de inventarios y gestión de laboratorios para la de la facultad de Ciencias.* Quito.

Cohn, M. (2004). *User stories applied: For agile software development.*

Creately. (18 de 10 de 2022). *La Guía Fácil de los Diagramas de Despliegue UML*. Recuperado el 11 de 10 de 2023, de https://creately.com/blog/es/diagramas/tutorial-de-diagrama-de-despliegue/#:~:text=Un%20diagrama%20de%20despliegue%20es,el%20middleware%20que%20los%20conecta.

Curbelo Carmona, M. (2013). *Herramienta web para la gestión de la base material de estudio en la Universidad de Pinar del Río "Hermanos Saíz Montes de Oca".* Pinar del Río.

de Souza, I. (14 de junio de 2019). *¿Qué es un servidor web y para qué sirve en Internet?* Recuperado el 18 de 9 de 2023, de https://rockcontent.com/es/blog/que-es-un-servidor/

Drupal. (2023). *Drupal.org*. Recuperado el 11 de 9 de 2023, de https://www.drupal.org/about

Durán, M. (2023). *Qué es la arquitectura en capas, ventajas y ejemplos*. Recuperado el 5 de 9 de 2023, de https://blog.hubspot.es/website/que-es-arquitectura-en-capas

Felipe. (3 de 9 de 2021). *Concepto de IDE y cuáles son sus características*. Recuperado el 19 de 9 de 2023, de https://www.hostingplus.pe/blog/concepto-de-ide-y-cuales-son-sus-caracteristicas/

Fernández Escribano, G. (2002). *Introducción a Extreme Programming.*

Gomez Mejia, M. (16 de 9 de 2013). *Patrones Arquitectónicos*. Recuperado el 30 de 10 de 2023, de https://ingeniods.wordpress.com/2013/09/16/patrones-arquitectonicos/

González Brito, H. R., Bauta Camejo, R. R., & Gainza Reyes, D. (2012). *EXTENSIÓN DEL ERP CEDRUX CON EL MARCO DE TRABAJO SAUXE. CASO DE ESTUDIO: SUBSISTEMA POSTGRADO 1.0.* La Habana, Cuba.

González, J., & Gaudioso, E. (2001). *Aprender y formar en Internet.*

Guerra, C. A. (2021). *Obtención de Requerimientos. Técnicas y Estrategia.* Recuperado el 6 de 8 de 2023, de https://sg.com.mx/revista/17/obtencion-requerimientos-tecnicas-y-estrategia

Hickson, I., & Hyatt, D. (2009). *HTML5*. Recuperado el 17 de junio de 2023, de http://dev.w3.org/html5/spec/Overview.html#html-vs-xhtml/

Hiken, A. (4 de 8 de 2023). *Una onza de prevención: seguridad y protección a través de estándares de codificación de software*. Recuperado el 13 de 9 de 2023, de https://es.parasoft.com/blog/an-ounce-of-prevention-software-safety-security-through-coding-standards/

Incorruptible. (2023). *Lista de chequeo: Ejemplo y guía de uso*. Recuperado el 15 de 9 de 2023, de https://incorruptible.mx/lista-de-chequeo-ejemplo-y-guia-de-uso/?expand_article=1

ISO/IEC 25010. (2022). *iso25000.com*. Recuperado el 22 de 9 de 2023, de https://iso25000.com/index.php/normas-iso-25000/iso-25010?start=3

Larman, C. (2005). *UML y Patrones.* Madrid.

Learning. (2023). *Linkedin*. Recuperado el 19 de 9 de 2023, de https://es.linkedin.com/learning/trucos-para-desarrollo-web/comillas-dobles-y-simples

Lucena, P. (2023). *¿Qué es el framework?* Recuperado el 18 de 9 de 2023, de https://www.cesuma.mx/blog/que-es-el-framework.html#:~:text=Un%20framework%20es%20un%20conjunto%20de%20reglas%20y%20convenciones%20que,utilizar%20como%20punto%20de%20partida.

Manz. (2021). *Indentación de código*. Obtenido de https://lenguajejs.com/fundamentos/introduccion/indentacion-de-codigo/indentacion/

MariaDB. (6 de 2 de 2023). *Notas de la versión de MariaDB 10.5.19*. Recuperado el 18 de 9 de 2023, de https://mariadb.com/kb/en/mariadb-10-5-19-release-notes/

Marín, R. (16 de 4 de 2019). *Los gestores de bases de datos más usados en la actualidad*. Recuperado el 18 de 9 de 2023, de https://www.inesem.es/revistadigital/informatica-y-tics/los-gestores-de-bases-de-datos-mas-usados/#:~:text=Un%20Sistema%20Gestor%20de%20Base,del%20modo%20m%C3%A1s%20eficiente%20posible.

Martínez Canelo, M. (24 de 6 de 2020). *¿Qué son los patrones de diseño de software?* Recuperado el 18 de 9 de 2023, de https://profile.es/blog/patrones-de-diseno-de-software/

Martínez de Sousa, J. (1990). *Diccionario de bibliología y ciencias afines*. Recuperado el 22 de junio de 2023, de https://dialnet.unirioja.es/servlet/libro?codigo=592296

MES. (2022). *Resolución No.47/22*. Recuperado el 22 de junio de 2023, de http://www.hospitalameijeiras.sld.cu/hha/sites/all/informacion/2022/Resolución47-2022.pdf

Molina Montero, B., Vite Cevallos, H., & Dávila Cuesta, J. (junio de 2018). *Metodologías ágiles frente a las tradicionales en el proceso de desarrollo de software.* Recuperado el 11 de 8 de 2023, de https://gc.scalahed.com/recursos/files/r161r/w25597w/438760423-269-823-1-PB-pdf.pdf

Ortega, C. (2023). *Cómo hacer un test de usabilidad*. Recuperado el 23 de 9 de 2023, de Questionpro: https://www.questionpro.com/blog/es/como-hacer-un-test-de-usabilidad/

Pérez, M. (2023). *Definición de Libro*. Recuperado el 22 de junio de 2023, de https://conceptodefinicion.de/libro/

PHP. (2023). *PHP.net*. Recuperado el 19 de 9 de 2023, de https://www.php.net/manual/es/intro-whatis.php

Pirani. (2014). *Checklist para la gestión de riesgos, según la ISO 31000*. Recuperado el 11 de 9 de 2023, de https://www.piranirisk.com/es/academia/especiales/checklist-para-la-gestion-de-riesgos-iso-31000

PostgreSQL. (s.f.). Recuperado el 17 de junio de 2023, de http://www.postgresql.org/about/

PRESSMAN, R. S. (2001). *Ingeniería del Software: una tecnología estratificada. Ingeniería del Software. Un enfoque prático.* (Quinta Edición ed.).

Qualitydevs. (5 de 8 de 2019). *¿Qué es Symfony? Y por qué es el mejor framework para crear aplicaciones web.* Recuperado el 19 de 9 de 2023, de https://www.qualitydevs.com/2019/08/05/que-es-symfony/

RAE. (2023). *Diccionario de la lengua española*. Recuperado el 22 de junio de 2023, de https://dle.rae.es/libro

Rehkopf, M. (2023). *Historias de usuario con ejemplos y plantilla*. Recuperado el 27 de 9 de 2023, de https://www.atlassian.com/es/agile/project-management/user-stories

ResourceMate. (2023). *ResourceMate® For Your Industry*. Recuperado el 16 de 8 de 2023, de https://www.resourcemate.com/

Rodríguez Salas, K. (2002). *Gestión de la información en las organizaciones*. Recuperado el 22 de junio de 2023, de https://www.revistas.una.ac.cr/index.php/bibliotecas/article/view/513

Sommerville, I. (2005). *Ingenieria del Software 7ma. Ed. -.* Madrid: Pearson Education, S.A.

Squirrels, J. (21 de 7 de 2023). *Todo sobre pruebas unitarias: técnicas, conceptos, práctica*. Recuperado el 14 de 9 de 2023, de https://codegym.cc/es/groups/posts/es.404.todo-sobre-pruebas-unitarias-tecnicas-conceptos-practica

Tasé Hernández, J., Cuba Puertas, I., & Rodríguez Montero, R. Y. (2005). *Sistema Integrado de Gestion Bibliotecaria. Modulos: Reproduccion y Digitalizacion de materiales. Proceso de Seleccion-Adquicicion de materiales.*

Tramullas, J. (1997). Los sistemas de información: una reflexión sobre información, sistema y documentación. *Revista General de Información y Documentación*, 207-229.

Vázquez, S. (2015). *¿Qué es Calibre? - Crea y Distribuye tu libro digital*. Recuperado el 22 de junio de 2023, de https://digital-editorial.com/que-es-calibre/

Vidal Ledo, M. J. (2012). Información y conocimiento para la dirección. *Revista Cubana de Educación Médica Superior*, 474-484 .

Wallace, D., Raggett, I., & Aufgang, J. (2003). *Extreme Programming for Web Projects.* Addison Wesley.

White, S. A. (2011). *Introduction to BPMN.*

Anexos

Anexo 1 Descripción de los roles y responsabilidades:

Tabla 36: Descripción de roles y responsabilidades

Roles	Responsabilidades
Administrador	Responsable del control de los usuarios, así como de la asignación de roles.
Vicedecana de formación	Máximo responsable de controlar el andar UCI de los estudiantes.
Supervisor del almacén	Encargado de realizar la entrega, recepción y control de los libros en el almacén.
Estudiantes y trabajadores (profesores y especialistas)	Personal de la facultad al que le es entregado los libros.

Anexo 2 Tarjetas CRC:

Tabla 37: Tarjeta CRC para la clase Taxonomía

Tarjeta CRC #2	
Clase: Taxonomía	
Responsabilidad	**Colaboración**
• Guardar y gestionar la información de una taxonomía. • Acceder a los términos de taxonomía asociados a la misma. • Disparar eventos relacionados con la creación y actualización de términos de taxonomía.	▪ Solicitud: Para acceder a los valores de las taxonomías asociadas a la solicitud. ▪ Libro: Para acceder a los valores de las taxonomías asociadas al libro.

Tabla 38: Tarjeta CRC para la clase EventSubscriber

Tarjeta CRC #3
Clase: EventSubscriber

Responsabilidad	Colaboración
• Suscribirse a eventos relacionados con la creación y actualización de nodos. • Acceder y manipular los valores de las taxonomías asociadas a la solicitud y el libro antes de que se guarde.	▪ Solicitud: Para acceder y manipular los valores de las taxonomías asociadas a la solicitud. ▪ Libro: Para acceder y manipular los valores de las taxonomías asociadas al libro.

Tabla 39: Tarjeta CRC para la clase Solicitud

Tarjeta CRC #4	
Clase: Solicitud	
Responsabilidad	**Colaboración**
• Guardar y gestionar la información vinculada a las solicitudes. • Mostrar detalles de una solicitud. • Disparar eventos relacionados con la creación y actualización de solicitudes.	• Taxonomía: Para acceder a los términos de taxonomía asociados a la solicitud. • Libro: Para registrar la bibliografía solicitada.

Tabla 40: Tarjeta CRC para la clase Usuario

Tarjeta CRC #5	
Clase: Usuario	
Responsabilidad	**Colaboración**
• Iniciar sesión, gestionar libros y solicitudes. • Guardar y gestionar la información vinculada al usuario.	• Taxonomía: Para acceder a los términos de taxonomía asociados al usuario.

Anexo 3 Entrevista:

Realizada a Dayanis Fundora Quintosa (supervisora y/o responsable del almacén de libros correspondiente a la Facultad 2 de la UCI)

Fecha: 10/05/2023

Objetivo: Comprender el funcionamiento del almacén de libros en cuanto a los servicios que allí se brindan.

1. ¿Descripción del flujo de pasos de los distintos procesos de préstamos y devoluciones?
2. ¿Cuáles son las reglas o normas que usted como encargada del almacén debe cumplir con respecto al servicio que se brinda?
3. ¿Cuáles son las principales deficiencias que usted como encargada opina que existen?

Anexo 4 Guía de observación del proceso:

Guía de Observación de los procesos del negocio

Como resultado de la entrevista realizada, surge esta guía de observación, que pretende describir cómo se llevan a cabo los procesos de entrada y salida de libros del almacén en la Facultad 2.

Criterios de observación:

- Actividades que se realizan.
- Relaciones que se establecen entre las actividades
- Roles que intervienen.
- Responsabilidades asociadas a los roles.
- Tiempo de duración de cada actividad.
- Principales dificultades presentadas.

Anexo 5 Acta de aceptación:

La Habana, de noviembre de 2023

"Año 65 de la Revolución"

Carta de Aceptación

Mediante la siguiente, hacemos constar que el estudiante Yadian Tresgallos González, en cumplimiento del desarrollo de la tesis "Sistema informático para la gestión de la bibliografía física docente en la Facultad 2 de la Universidad de las Ciencias Informáticas", hace entrega del producto (aplicación web).

El cliente ha revisado y probado el software, por lo que pudo comprobar que los requerimientos del software se implementaron en su totalidad. El producto entregado, contribuirá sin duda alguna, a la gestión de los textos en la Facultad 2 de la Universidad de las Ciencias Informáticas.

Conforme con lo anterior se expresa la aceptación del producto entregado.

FACULTAD 2 UCI Universidad de las Ciencias Informáticas DECANATO

Dayanis Fundora Quintosa (supervisora y/o responsable del almacén de libros)

Anexo 6 Pruebas de aceptación:

Tabla 41: Clases de equivalencia para la TI "Editar solicitud"

Identificador	Entrada (clase)	Clases válidas	Clases inválidas
Nombre	Nombre	Nombre = a-z	Nombre = caracteres especiales
Primer apellido	Apellido	Apellido = a-z	Apellido = caracteres especiales
Segundo apellido	Apellido	Apellido = a-z	Apellido = caracteres especiales
Solapín	Solapín	Solapín = 0-9	Solapín = caracteres especiales
Categoría	Categoría	Categoría = seleccionado	Categoría = sin selección
Nombre del libro	Título	Título = seleccionado	Título = sin selección
Año	Año	Año = seleccionado	Año = sin selección
Semestre	Semestre	Semestre = seleccionado	Semestre = sin selección

Tabla 42: Caso de prueba para la TI "Editar solicitud"

<table>
<tr><td>Escenario: Operación de Editar.</td><td rowspan="2">Descripción: Selecciona la opción Editar.</td><td>Respuesta del sistema</td><td>Flujo central</td></tr>
<tr><td>Identificadores</td><td rowspan="9">Brinda la posibilidad de editar los siguientes datos:
Nombre
Primer apellido
Segundo apellido
Solapín
Categoría
Nombre del libro
Año
Semestre</td><td rowspan="9">/node/edit</td></tr>
<tr><td>Nombre</td><td>-</td></tr>
<tr><td>Primer apellido</td><td>-</td></tr>
<tr><td>Segundo apellido</td><td>-</td></tr>
<tr><td>Solapín</td><td>-</td></tr>
<tr><td>Categoría</td><td>-</td></tr>
<tr><td>Nombre del libro</td><td>-</td></tr>
<tr><td>Año</td><td>-</td></tr>
<tr><td>Semestre</td><td>-</td></tr>
<tr><td>Escenario: Opción de Guardar.</td><td rowspan="2">Descripción: Modifica y/o selecciona los datos de una solicitud y selecciona la opción Guardar.</td><td>Respuesta del sistema</td><td>Flujo central</td></tr>
<tr><td>Identificadores</td><td rowspan="8">Valida los datos, modifica la solicitud, muestra una vista previa de la solicitud modificada y muestra un mensaje de información (Mensaje de estado: Solicitud Román Tresgallos González ha sido actualizada.</td><td rowspan="8">/node/edit</td></tr>
<tr><td>Nombre</td><td>Román</td></tr>
<tr><td>Primer apellido</td><td>Tresgallos</td></tr>
<tr><td>Segundo apellido</td><td>González</td></tr>
<tr><td>Solapín</td><td>E-18480</td></tr>
<tr><td>Categoría</td><td>Estudiante</td></tr>
<tr><td>Nombre del libro</td><td>Matemática Discreta</td></tr>
<tr><td>Año</td><td>5</td></tr>
<tr><td>Semestre</td><td>Décimo</td></tr>
<tr><td>Escenario: Opción de Guardar con datos incompletos.</td><td rowspan="2">Descripción: Modifica y/o selecciona los datos de una solicitud, dejando datos vacíos y selecciona la opción Guardar.</td><td>Respuesta del sistema</td><td>Flujo central</td></tr>
<tr><td>Identificadores</td><td></td><td></td></tr>
</table>

Nombre	Román	Muestra un indicador sobre los campos vacíos y muestra un mensaje de información (**Mensaje de error:** El campo es obligatorio).	/node/edit
Primer apellido	Tresgallos		
Segundo apellido	-		
Solapín	E-		
Categoría	Estudiante		
Nombre del libro	Matemática Discreta		
Año	5		
Semestre	Décimo		
Escenario: Opción de Regresar al sitio.	**Descripción:** Selecciona la opción de Regresar al sitio.	**Respuesta del sistema**	**Flujo central**
Identificadores			
Nombre	Yadian	Elimina los datos creados y regresa a la página anterior.	/node/edit
Primer apellido	Tresgallos		
Segundo apellido	González		
Solapín	E-18480		
Categoría	Estudiante		
Nombre del libro	Matemática Discreta		
Año	5		
Semestre	Décimo		

Tabla 43: Clases de equivalencia para la TI "Añadir libro"

Identificador	Entrada (clase)	Clases válidas	Clases inválidas
Nombre	Nombre	Nombre = a-z	Nombre = caracteres especiales
Existencia Total	Cantidad	Cantidad = 0-9	Cantidad = caracteres especiales

Tabla 44: Caso de prueba para la TI "Añadir libro"

Escenario: Operación de Añadir libro.	**Descripción:** Selecciona la opción Añadir libro.	**Respuesta del sistema**	**Flujo central**
Identificadores			

Nombre	-	Brinda la posibilidad de introducir o seleccionar de manera obligatoria los siguientes datos: • Nombre • Existencia total	/node/add/libro
Existencia total	-		
Escenario: Opción de Guardar. **Identificadores**	**Descripción:** Introduce los datos de un libro y selecciona la opción Guardar.	**Respuesta del sistema**	**Flujo central**
Nombre	Programación Web	Valida los datos, crea un nuevo libro, muestra una vista previa del libro creado y muestra un mensaje de información (**Mensaje de estado:** Libro *Programación Web* se ha creado.	/node/add/libro
Existencia total	300		
Escenario: Opción de Guardar con datos incompletos. **Identificadores**	**Descripción:** Introduce alguno de los datos de un libro y selecciona la opción Guardar.	**Respuesta del sistema**	**Flujo central**
Nombre	Programación Web	Muestra un indicador sobre los campos vacíos y muestra un mensaje de información (**Mensaje de error:** El campo es obligatorio).	/node/add/libro
Existencia total	-		
Escenario: Opción de Regresar al sitio. **Identificadores**	**Descripción:** Selecciona la opción de Regresar al sitio.	**Respuesta del sistema**	**Flujo central**
Nombre	Programación Web	Elimina los datos creados y regresa a la página anterior.	/node/add/libro
Existencia total	300		

Tabla 45: Clases de equivalencia para la TI "Editar libro"

Identificador	Entrada (clase)	Clases válidas	Clases inválidas
Nombre	Nombre	Nombre = a-z	Nombre = caracteres especiales
Existencia Total	Cantidad	Cantidad = 0-9	Cantidad = caracteres especiales

Tabla 46: Caso de prueba para la TI "Editar libro"

Escenario: Operación de Editar.	**Descripción:** Selecciona la opción Editar.	**Respuesta del sistema**	**Flujo central**
Identificadores			
Nombre	Programación Web	Brinda la posibilidad de editar los siguientes datos: • Nombre • Existencia total	/node/edit
Existencia total	300		
Escenario: Opción de Guardar.	**Descripción:** Modifica los datos de un libro y selecciona la opción Guardar.	**Respuesta del sistema**	**Flujo central**
Identificadores			
Nombre	Programación Web	Valida los datos, modifica los datos del libro, muestra una vista previa del libro modificado y muestra un mensaje de información (**Mensaje de estado:** Libro *Programación Web* ha sido actualizado.	/node/edit
Existencia total	450		
Escenario: Opción de Guardar con datos incompletos.	**Descripción:** Modifica los datos de un libro, dejando campos vacíos y selecciona la opción Guardar.	**Respuesta del sistema**	**Flujo central**
Identificadores			
Nombre	Programación Web	Muestra un indicador sobre los campos vacíos	/node/edit
Existencia total	-		

		y muestra un mensaje de información (**Mensaje de error:** El campo es obligatorio).	
Escenario: Opción de Regresar al sitio.	**Descripción:** Selecciona la opción de Regresar al sitio.	**Respuesta del sistema**	**Flujo central**
Identificadores			
Nombre	Programación Web	Elimina los datos creados y regresa a la página anterior.	/node/edit
Existencia total	450		

Tabla 47: Clases de equivalencia para la TI "Añadir usuario"

Identificador	Entrada (clase)	Clases válidas	Clases inválidas
Dirección de correo electrónico	Correo	Correo = a-z, 0-9	Correo = vacía
Nombre de usuario	Nombre	Nombre = a-z, 0-9	Nombre = vacía
Contraseña	Contraseña	Contraseña = a-z, 0-9	Contraseña = vacía
Rol	Rol	Rol = seleccionado	Rol = sin selección

Tabla 48: Caso de prueba para la TI "Añadir usuario"

Escenario: Operación de Añadir usuario.	**Descripción:** Selecciona la opción Añadir usuario.	**Respuesta del sistema**	**Flujo central**
Identificadores			
Dirección de correo electrónico	-	Brinda la posibilidad de introducir o seleccionar de manera obligatoria los siguientes datos: • Dirección de correo electrónico • Nombre de usuario • Contraseña • Rol	/admin/people/create
Nombre de usuario	-		
Contraseña	-		
Rol	-		

Escenario: Opción de Crear nueva cuenta.	**Descripción:** Introduce los datos de un usuario y selecciona la opción Crear nueva cuenta.	**Respuesta del sistema**	**Flujo central**
Identificadores			
Dirección de correo electrónico	yadian.tres@uci.cu	Valida los datos, crea un nuevo usuario, muestra una vista previa del usuario creado y muestra un mensaje de información (**Mensaje de estado:** Creada una nueva cuenta de usuario para *Yadian*.	/admin/people/create
Nombre de usuario	Yadian		
Contraseña	Yadian123		
Rol	Administrador		
Escenario: Opción de Crear nueva cuenta con datos incompletos.	**Descripción:** Introduce alguno de los datos de un usuario y selecciona la opción Crear nueva cuenta.	**Respuesta del sistema**	**Flujo central**
Identificadores			
Dirección de correo electrónico	yadian.tres@uci.cu	Muestra un indicador sobre los campos vacíos y muestra un mensaje de información (**Mensaje de error:** El campo es obligatorio).	/admin/people/create
Nombre de usuario			
Contraseña			
Rol	Administrador		
Escenario: Opción de Regresar al sitio.	**Descripción:** Selecciona la opción de Regresar al sitio.	**Respuesta del sistema**	**Flujo central**
Identificadores			
Dirección de correo electrónico	yadian.tres@uci.cu	Elimina los datos creados y regresa a la página anterior.	/admin/people/create
Nombre de usuario	Yadian		
Contraseña	Yadian123		
Rol	Administrador		

Tabla 49: Clases de equivalencia para la TI "Editar usuario"

Identificador	Entrada (clase)	Clases válidas	Clases inválidas
Dirección de correo electrónico	Correo	Correo = a-z, 0-9	Correo = vacía
Nombre de usuario	Nombre	Nombre = a-z, 0-9	Nombre = vacía
Contraseña	Contraseña	Contraseña = a-z, 0-9	Contraseña = vacía
Rol	Rol	Rol = seleccionado	Rol = sin selección

Tabla 50: Caso de prueba para la TI "Editar usuario"

Escenario: Operación de Editar.	**Descripción:** Selecciona la opción Editar.	**Respuesta del sistema**	**Flujo central**
Identificadores			
Dirección de correo electrónico	yadian.tres@uci.cu	Brinda la posibilidad de editar los siguientes datos: • Dirección de correo electrónico • Nombre de usuario • Contraseña • Rol	/user/edit
Nombre de usuario	Yadian		
Contraseña	Yadian123		
Rol	Administrador		
Escenario: Opción de Guardar.	**Descripción:** Modifica los datos de un usuario y selecciona la opción Guardar.	**Respuesta del sistema**	**Flujo central**
Identificadores			
Dirección de correo electrónico	yadian.tres@uci.cu	Valida los datos, modifica los datos del usuario, muestra una vista previa del usuario modificado y muestra un mensaje de información (**Mensaje de estado:** Se han guardado los cambios.	/user/edit
Nombre de usuario	Yadian		
Contraseña	Tresgallos333		
Rol	Administrador		

Escenario: Opción de Guardar con datos incompletos.	**Descripción:** Modifica los datos de un usuario, dejando campos vacíos y selecciona la opción Guardar.	**Respuesta del sistema**	**Flujo central**
Identificadores			
Dirección de correo electrónico	yadian.tres@uci.cu	Muestra un indicador sobre los campos vacíos y muestra un mensaje de información (**Mensaje de error:** El campo es obligatorio).	/user/edit
Nombre de usuario	Yadian		
Contraseña	-		
Rol	Administrador		
Escenario: Opción de Regresar al sitio.	**Descripción:** Selecciona la opción de Regresar al sitio.	**Respuesta del sistema**	**Flujo central**
Identificadores			
Dirección de correo electrónico	yadian.tres@uci.cu	Elimina los datos creados y regresa a la página anterior.	/user/edit
Nombre de usuario	Yadian		
Contraseña	Tresgallos333		
Rol	Administrador		