**TÍTULO DEL PROYECTO**

**MARIÑOTECA**

**NOMBRES DE LOS ESTUDIANTES**

**Luis Fernando Pérez Salamanca**

**Adrián Daniel Ustariz Ortiz**

**Laura Andrea Carrillo algo**

**Erik Sneyder Plata Arias**

**Luis Felipe Velandia Llanes**

**GRUPO ASIGNADO**

**U2**

**DOCENTES INVOLUCRADOS**

**Juan Carlos Mariño**

**CAMPUSLANDS**

**SALON Sputnik  
RUTA Java**

**FLORIDABLANCA**

**2024**

**TÍTULO DEL PROYECTO**

1. **SITUACIÓN PROBLEMA**

La Biblioteca Mariñoteca busca modernizar su sistema de gestión de libros y usuarios para mejorar la eficiencia en el registro, préstamo y devolución de libros. El objetivo es desarrollar un sistema CRUD que permita a los administradores y clientes realizar estas tareas de manera eficiente y segura.

**1.2 Objetivos del Proyecto**

* **Mejorar la eficiencia: Reducir el tiempo necesario para registrar y buscar libros.**
* **Optimizar la gestión: Facilitar la administración de usuarios y reservas.**
* **Proporcionar una mejor experiencia: Ofrecer una interfaz intuitiva y accesible para los usuarios.**

**1.3 Justificación y Alcance**

La digitalización de los procesos de la biblioteca reducirá los errores humanos y permitirá un acceso más rápido y sencillo a la información. El sistema abarcará desde el registro y la gestión de usuarios hasta la administración completa de los recursos de la biblioteca.

1. **LEVANTAMIENTO DE REQUERIMIENTOS**

Para abordar esta situación, se levantaron los siguientes requerimientos en base a las funciones proporcionadas en el archivo marinoteca.py y el archivo de datos data.json.

1. **REQUERIMIENTOS**
   1. **Requerimientos Funcionales**

**Gestión de Usuarios:**

* **Registro de Usuarios:** El sistema debe permitir el registro de nuevos usuarios con información básica (nombres, apellidos, número de identificación, dirección, número de teléfono móvil y contraseña).
* **Eliminación de Usuarios:** Los administradores deben poder eliminar usuarios del sistema.
* **Autenticación de Usuarios:** Los usuarios deben poder iniciar sesión con su número de identificación y contraseña.
* **Autenticación de Administradores:** Los administradores deben poder iniciar sesión con su usuario y contraseña para acceder a funcionalidades adicionales.

**Gestión de Libros:**

* **Agregar Libros:** Los administradores deben poder registrar nuevos libros en el sistema proporcionando título, autor, categoría, fecha de publicación y cantidad.
* **Eliminar Libros:** Los administradores deben poder eliminar libros del sistema, ya sea totalmente o reduciendo la cantidad disponible.
* **Agregar Copias de Libros:** Los administradores deben poder aumentar la cantidad de copias disponibles de un libro existente.
* **Mostrar Todos los Libros:** El sistema debe permitir a los administradores y usuarios visualizar la lista completa de libros disponibles.
* **Mostrar Libros por Género y Autor:** El sistema debe permitir la búsqueda y visualización de libros filtrados por género y autor.

**Reserva y Devolución de Libros:**

* **Reservar Libros:** Los usuarios deben poder reservar libros disponibles.
* **Devolución de Libros:** El sistema debe permitir la devolución de libros prestados y actualizar la disponibilidad del inventario.
  1. **Requerimientos No Funcionales**

**Diseño e Interfaz de Usuario:**

* **Interfaz Intuitiva: El sistema debe tener una interfaz intuitiva y fácil de usar para todas las funciones, tanto para administradores como para usuarios.**
* **Claridad en los Mensajes: Todos los mensajes y prompts del sistema deben ser claros y proporcionar suficiente información para guiar al usuario en sus acciones.**

**Rendimiento y Escalabilidad:**

* **Tiempo de Respuesta: El sistema debe ser capaz de manejar solicitudes de múltiples usuarios sin demoras significativas.**
* **Escalabilidad: El sistema debe ser capaz de escalar para manejar un número creciente de usuarios y libros sin degradación del rendimiento.**

**Seguridad:**

* **Protección de Datos: La información de los usuarios, incluyendo las contraseñas, debe estar adecuadamente protegida.**
* **Control de Acceso: Las funciones críticas del sistema deben estar protegidas y accesibles solo para los administradores autorizados.**

**Mantenimiento y Actualización:**

* **Documentación Clara: El código debe estar bien documentado para facilitar el mantenimiento y futuras actualizaciones.**
* **Gestión de Errores: El sistema debe manejar adecuadamente los errores y proporcionar mensajes de error útiles para los usuarios.**

**DESCRIPCION DE FUNCIONES PRINCIPALES**

**leer\_datos\_biblioteca(): Lee los datos de la biblioteca desde un archivo JSON.**

**guardar\_datos\_biblioteca(data): Guarda los datos actualizados de la biblioteca en un archivo JSON.**

**clear(): Limpia la consola.**

**wait\_for\_keypress(): Espera hasta que el usuario presione una tecla para continuar.**

**generadorID(data): Genera un nuevo ID para un libro basándose en el último ID usado.**

**main\_principal\_ingreso(): Función principal para manejar el ingreso de usuarios y administradores.**

**security\_admin(): Maneja la autenticación de administradores.**

**security\_cliente(): Maneja la autenticación de clientes.**

**main\_admin\_biblioteca(): Menú principal para las operaciones del administrador.**

**registrar\_usuario(): Registra nuevos usuarios.**

**eliminar\_usuario(): Elimina usuarios existentes.**

**agregar\_libro(): Agrega nuevos libros a la biblioteca.**

**eliminar\_libro(): Elimina libros existentes de la biblioteca.**

**mostrar\_todos(), mostrar\_genero(), mostrar\_autor(), mostrar\_disponibles(), clientes\_con\_libros(), reservar\_manualmente(), devolver\_libro(): Funciones auxiliares para mostrar libros y gestionar reservas y devoluciones.**

1. **HISTORIAS DE USUARIO CON CRITERIOS DE ACEPTACIÓN**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | | | |
| **HISTORIA DE USUARIO** | | | | |
| **Prioridad: Alta** | | | | |
| **CÓDIGO DEL**  **REQUERIMIENTO:** | | RF01 | **Actor** | Usuario |
| **NOMBRE DEL**  **REQUERIMIENTO** | | **Registro de usuarios** | | |
| **Descripción** | | | | |
| Como usuario, quiero registrarme en el sistema proporcionando mi información personal para poder usar la biblioteca y acceder al programa completo. | | | | |
| **Funcionalidad** | | | | |
| Permitir a los usuarios registrarse proporcionando su información personal de forma que por medio de credenciales como usuario y contraseña se pueda acceder al programa de forma optima y este pueda acceder al resto de funcionalidades disponibles en su rol. | | | | |
| **Criterios de aceptación** | 1.El usuario puede registrarse proporcionando nombres, apellidos, número de identificación, dirección, número de teléfono móvil y contraseña.  2.El sistema muestra un mensaje de confirmación después del registro exitoso.  3.Se despliega el menú con acceso garantizado al programa en su versión disponible para el usuario. | | | |
| **Restricciones** | | | | |
| El registro debe ser único por número de identificación para que de esta forma no se reescriban los datos y así no generar fallas a largo plazo. | | | | |
| **HISTORIA DE USUARIO 2** | | | | |
| **Prioridad: Alta** | | | | |
| **CÓDIGO DEL**  **REQUERIMIENTO:** | | RF03 | **Actor** | Administrador |
| **NOMBRE DEL**  **REQUERIMIENTO** | | **Autenticación de Administradores** | | |
| **Descripción** | | | | |
| Como administrador, quiero iniciar sesión con mi usuario y contraseña para acceder a funcionalidades administrativas y tener el control total del programa al momento de ejecutarse, poder usar cada una de las funciones establecidas. | | | | |
| **Funcionalidad** | | | | |
| Permitir a los administradores iniciar sesión para acceder a funciones adicionales y asi obtener manejo amplio del sistema en base a su rol. | | | | |
| **Criterios de aceptación** | 1. El administrador puede iniciar sesión con credenciales válidas siempre que concuerden con las que la base de datos tiene predefinidas para el rol.  2. El sistema muestra un mensaje de error si las credenciales son incorrectas.  3.El sistema muestra el menú principal con las funciones de usuario y de administrador, despliega un total acceso para el rol y así tener el control total del programa. | | | |
| **Restricciones** | | | | |
| Solo los administradores con credenciales válidas pueden acceder a las funciones administrativas. | | | | |
| **HISTORIA DE USUARIO 3** | | | | |
| **Prioridad: Alta** | | | | |
| **CÓDIGO DEL**  **REQUERIMIENTO:** | | RF05 | **Actor** | Administrador |
| **NOMBRE DEL**  **REQUERIMIENTO** | | **Gestión de Libros - Agregar Libros** | | |
| **Descripción** | | | | |
| Como administrador, quiero agregar nuevos libros proporcionando la información necesaria para mantener actualizado el catálogo de la biblioteca y que por consecuente el programa pueda mostrar todo lo relacionado a la biblioteca con la información actualizada para el usuario, facilitando su uso. | | | | |
| **Funcionalidad** | | | | |
| Permitir a los administradores registrar nuevos libros en el sistema en base a los requisitos de clasificación de libros que tiene la biblioteca dentro del programa. | | | | |
| **Criterios de aceptación** | 1. El administrador puede agregar un nuevo libro a la base de datos del programa proporcionando título, autor, categoría, fecha de publicación y cantidad.  2. El sistema actualiza el catálogo de libros con la nueva información. | | | |
| **Restricciones** | | | | |
| El libro agregado debe tener todos los campos obligatorios completos. | | | | |
|  | | | | |

1. **METODOLOGÍA**

Se seguirá la metodología Scrum para el desarrollo del proyecto. Las actividades y tareas se realizarán en ciclos de sprints, con las siguientes fases y roles:

**Roles:**

1. **Product Owner:**  Responsable de definir y priorizar los requerimientos.
2. **Scrum Master:** Responsable de facilitar el proceso Scrum y eliminar impedimentos.
3. **Equipo de Desarrollo:** Responsable de implementar y entregar las funcionalidades acordadas en cada sprint.

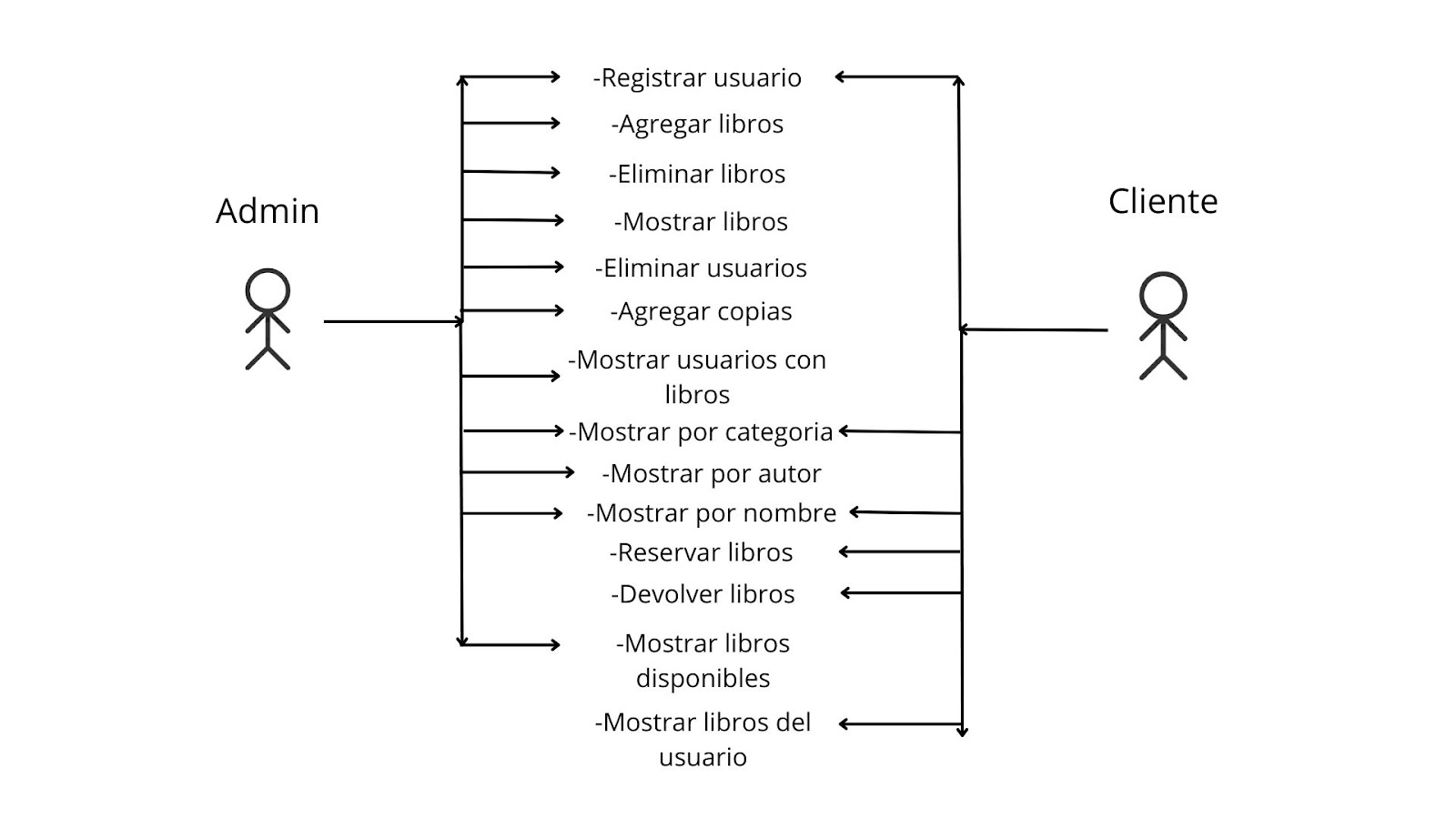
**Ciclos de Trabajo:**

1. **Sprint Planning:** Planificación del trabajo a realizar en el sprint.
2. **Daily Scrum:** Reuniones diarias para revisar el progreso y coordinar el trabajo.
3. **Sprint Review:** Revisión del trabajo completado al final de cada sprint.
4. **Sprint Retrospective:** Reflexión sobre el sprint para identificar mejoras en el proceso.
5. **EVIDENCIA DE PLANTEAMIENTO DE PLATAFORMA DE TRABAJO**

Se utilizará una plataforma de gestión de proyectos, en este caso es Jira, esto con el propósito general de gestionar las historias de usuario, tareas y seguimiento del progreso del equipo. Se incluirán capturas de pantalla y descripciones detalladas de cómo se configuró y utilizó la plataforma para el proyecto.

**7. Diagrama de Casos de Uso**

**DIAGRAMA DE CASOS DE USO**



**Administrador**

En el diagrama de casos de uso podemos evidenciar que el administrador de la biblioteca es el encargado de hacer todo lo relacionado con la gestión, control y organización de tanto usuario como libros. Este posee control total sobre la base de datos simulada en un archivo .json, sin embargo, también se otorga el poder al usuario de que el mismo realice un registro manual, o sea, tiene acceso parcial al archivo json, con el único fin de generar registro de usuarios. El administrador tiene la posibilidad de llevar un control absoluto de todos los libros que se posee en la base de datos, esto quiere decir que puede buscar libros por categorías, nombres, libros disponibles y también la posibilidad de sumar un número de cantidad ilimitadas, esto quiere decir que al momento de eliminar libros también se pueden eliminar cantidades específicas elegidas por el administrador.

El administrador también tiene control absoluto sobre los usuarios, teniendo opciones para agregar y eliminar usuarios, al momento de agregar usuarios a estos se les genera una lista como atributo con el fin de añadir libros a esta lista, o sea, cuando estos quieran añadir tomos de libros y demás, se añadirán a esta lista.

**Usuario**

El usuario tiene las funcionalidades de buscar libros por categorías, nombre, y disponibilidad, también la posibilidad de reservar libros y devolverlos y un acceso a un perfil donde puede ver su nombre y los libros que actualmente posee.

El usuario tiene la posibilidad de buscar libros con el fin de que este pueda inmediatamente revisar si el libro en el que este esta interesado se encuentra disponible, o incluso si este se encuentra indeciso, poder ver libros por categorías o disponibilidad con el fin de tomar una decisión acerca de qué libro puede llevarse.

La funcionalidad de poder reservar libros tiene su condición, el usuario puede reservar el libro que desee, sin embargo, no puede reservar el mismo libro dos veces, esto quiere decir, que si el usuario ya posee x libro, no puede reservar una segunda copia de este, esto con el fin de poder llevar un mejor control de los libros que reservan los usuarios.

La función de devolución funciona eliminando el libro de la lista del usuario para posteriormente pasar este tomo a la biblioteca, de momento no se encuentra ninguna condición de multas por retardo para el usuario, esto quiere decir que el usuario puede devolver el libro cuando desee.

Por último la revisión del perfil del usuario únicamente da la posibilidad de que el usuario pueda revisar los libros que actualmente posee, también puede ver su nombre, y sus datos básicos, con el fin de que el usuario tenga la posibilidad de tener un mínimo de control dentro de su perfil, brindando así, una experiencia de usuario mucho mas cómoda y personalizable.

**8. Diagrama de Clases**

(Incluir un diagrama de clases detallando las entidades y sus relaciones)

**9. Diseño de la Base de Datos**

Una base de datos es un sistema organizado para almacenar, gestionar y

recuperar datos de manera eficiente. En este proyecto, se utiliza JSON (JavaScript

Object Notation) como formato de almacenamiento de la base de datos. JSON es

un formato ligero y de fácil uso para el intercambio y almacenamiento de datos.

Las funciones JSON, presentadas por primera vez en SQL Server 2016 (13.x),

permiten combinar conceptos NoSQL y relacionales en la misma base de datos.

Puedes combinar columnas relacionales clásicas con columnas que contienen

documentos con formato de texto JSON en la misma tabla, analizar e importar

documentos JSON en estructuras relacionales o dar formato de texto JSON a

datos relacionales.

**9.1 Modelo Entidad-Relación (ERD)**

(Incluir el diagrama ERD con todas las entidades y relaciones)

**9.2 Descripción de las Tablas y Relaciones**

* **Tabla Usuarios:** Almacena la información de los usuarios.
* **Tabla Libros:** Almacena la información de los libros.
* **Tabla Reservas:** Almacena la información de las reservas de libros.

**10. Diseño de la Interfaz de Usuario**

**10.1 Prototipos de Pantalla**



**10.2 Descripción de la Navegación**

* **Pantalla de Inicio:** Presenta opciones para iniciar sesión o registrarse, también funciones para ingresar como administrador y salir del programa.
* **Pantalla de Registro:** Permite a los usuarios registrarse ingresando su información personal.
* **Pantalla de Inicio de Sesión:** Permite a los usuarios iniciar sesión con su número de identificación y contraseña.
* **Panel de Administración:** Acceso a funcionalidades adicionales para administradores.
* **Pantalla de Búsqueda de Libros:** Permite a los usuarios buscar libros por diferentes criterios.
* **Pantalla de Reservas:** Permite a los usuarios ver y gestionar sus reservas.
* **10.3 Detalles de Diseño de la UI/UX**
* **Colores y Tipografías:** Se utiliza una gama de colores únicamente relacionadas al azul y al negro puesto que representan confiabilidad y comodidad a la hora de leer, temas que por supuesto se relacionan a una biblioteca, son adecuados y agradables a la vista del usuario
* **Elementos de Interfaz:** Botones, formularios, mensajes de error y confirmación.

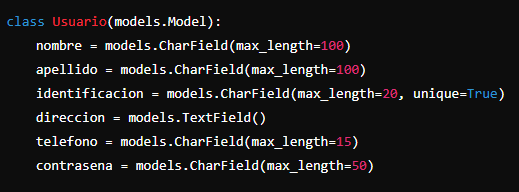
**11. Descripción de Funcionalidades**

* **11.1 Registro de Usuarios**
* **Descripción:** Permite el registro de nuevos usuarios en el sistema.
* **Flujo de Trabajo:**
  1. El usuario accede a la pantalla de registro.
  2. Completa el formulario con su información personal.
  3. Envía el formulario.
  4. El sistema verifica los datos y registra al usuario.
* **11.2 Inicio de Sesión**
* **Descripción:** Permite a los usuarios autenticarse en el sistema.
* **Flujo de Trabajo:**
  1. El usuario accede a la pantalla de inicio de sesión.
  2. Ingresa su número de identificación y contraseña.
  3. El sistema verifica las credenciales y permite el acceso.

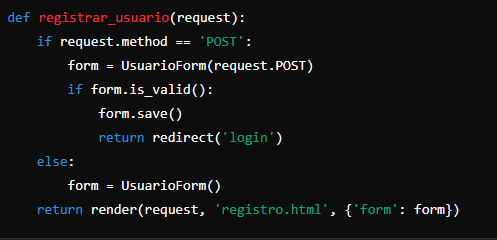
**12. Implementación**

* **12.1 Detalles sobre el Entorno de Desarrollo**
* **Lenguaje de Programación:** Python.
* **Base de Datos:** Almacenaje local en archivos .JSON.
* **12.2 Ejemplos de Código**

**Modelo de Usuario:**



**Vista de Registro de Usuario:**



**13. Pruebas del Sistema**

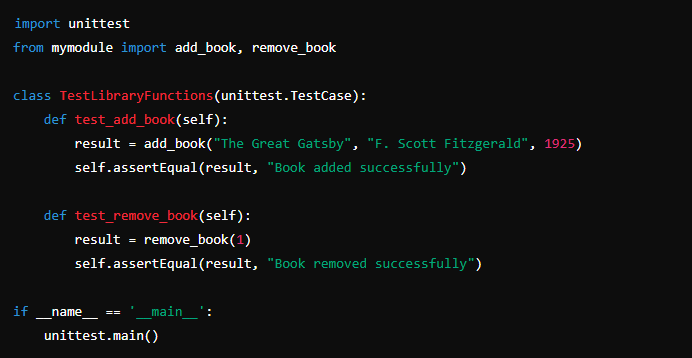
**13.1 Plan de Pruebas**

El plan de pruebas del proyecto "Mariñoteca" abarca varias estrategias para asegurar la calidad y funcionalidad del sistema. Las pruebas se dividen en tres categorías principales: pruebas unitarias, pruebas de integración y pruebas de aceptación.

* **13.2 Casos de Prueba**

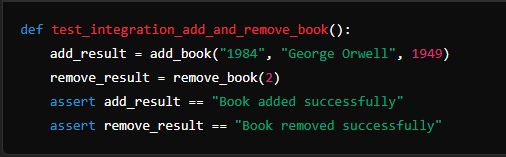
**Prueba unitaria:**

**Descripción: Las pruebas unitarias se centran en verificar la funcionalidad de componentes individuales del sistema, como funciones y métodos. Cada unidad de código se prueba de forma aislada para asegurar que funcione correctamente.**



**Prueba de integración:**

**Descripción: Las pruebas de integración se realizan para verificar la interacción entre diferentes módulos y componentes del sistema. Se aseguran de que las unidades individuales trabajen juntas correctamente.**



**Pruebas de aceptación:**

**Descripción: Las pruebas de aceptación se realizan para asegurar que el sistema cumple con los requisitos del cliente y que las funcionalidades principales funcionan según lo esperado. Estas pruebas son generalmente realizadas por el equipo de calidad o por los usuarios finales.**

Característica: Gestión de libros

Escenario: agregar un libro nuevo

Dado que he iniciado sesión como administrador

Cuando agrego un libro con título "Un mundo feliz" y autor "Aldous Huxley"

Entonces debería ver "Libro agregado correctamente".

Escenario: quitar un libro

Dado que he iniciado sesión como administrador

Cuando quito el libro con ID 3

Entonces debería ver "Libro eliminado correctamente".

* **Prueba de Registro de Usuario:**

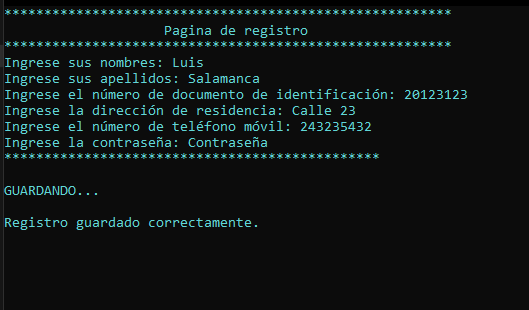
**Descripción: Verificar que el sistema permite registrar un nuevo usuario correctamente.**

**Pasos:**

* + 1. **Acceder a la página de registro.**
    2. **Completar el formulario de registro con datos válidos.**
    3. **Enviar el formulario.**
    4. **Verificar que el usuario es redirigido a la página de inicio de sesión con un mensaje de éxito.**

**Resultados Esperados: El usuario es registrado correctamente y puede iniciar sesión.**

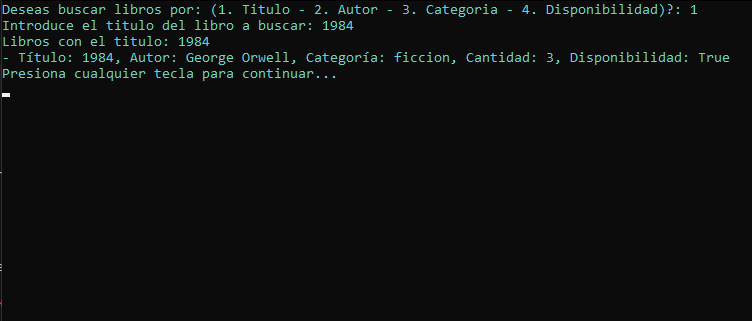
**13.3 Resultados de las Pruebas**



**Prueba de Búsqueda de Libros:**

* **Descripción:** Verificar que los usuarios pueden buscar libros por título, autor o año de publicación.
* **Pasos:**
  1. Acceder a la página de búsqueda de libros.
  2. Ingresar "1984" en el campo de búsqueda de título.
  3. Hacer clic en el botón de búsqueda.
  4. Verificar que el libro "1984" de George Orwell aparece en los resultados.
* **Resultados Esperados:** El libro buscado aparece correctamente en los resultados.





**Resultados de las Pruebas**

Proporcionar un resumen de los resultados obtenidos en las pruebas, incluyendo capturas de pantalla de las pruebas realizadas, logs de errores encontrados y cómo fueron corregidos.

**Ejemplo de Resultados:**

* **Prueba de Registro de Usuario:**

Resultado: Éxito

* **Prueba de Búsqueda de Libros:**

Resultado: Éxito

**14. Metodología de Desarrollo**

**17. Conclusiones**

* **17.1 Logros del Proyecto**

Cumplimiento de Requisitos: El desarrollo del software para la biblioteca Mariñoteca ha sido exitoso al cumplir con todos los requisitos funcionales y no funcionales especificados al inicio del proyecto. Cada una de las funciones esperadas ha sido implementada de manera completa y efectiva.

Mejora Continua y Adaptabilidad: A través del uso de la metodología Scrum, el equipo ha demostrado una excelente capacidad de adaptación y mejora continua. Las iteraciones y sprints permitieron ajustar y refinar las funcionalidades de Mariñoteca, asegurando que el producto final cumpla con las expectativas del cliente y usuarios finales.

Documentación Completa: El proyecto incluyó una documentación exhaustiva que facilita el mantenimiento y la futura expansión del sistema. La documentación abarca tanto los aspectos técnicos del desarrollo como las guías para los usuarios, asegurando una transición fluida para futuros desarrolladores y usuarios.

Satisfacción del Usuario: Las pruebas realizadas y la retroalimentación obtenida sugieren que los usuarios están satisfechos con las funcionalidades y el rendimiento de la biblioteca Mariñoteca. El enfoque centrado en el usuario y la iteración constante contribuyeron a crear una solución que cumple con las necesidades reales de los usuarios.

**Evidencia Jira:**

