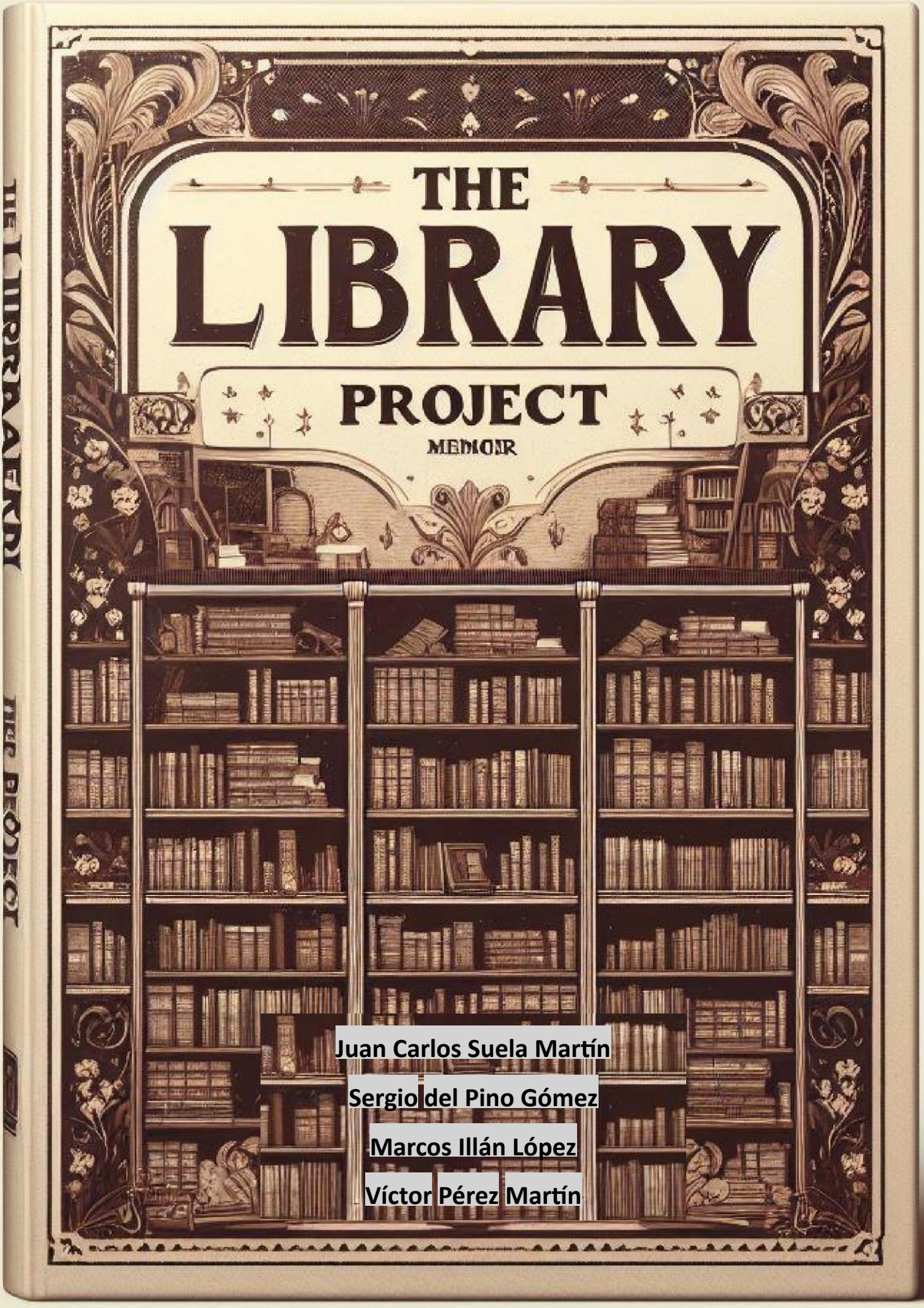
THE
LIBRARY
PROJECT
MEMOIR
Juan Carlos Suela Martín
Sergio del Pino Gómez
Marcos Illán López
Víctor Pérez Martín

Contenido

Miembros del Proyecto:

El equipo de trabajo de la empresa está formado por los siguientes miembros:

- Juan Carlos Suela Martín
- Marcos Illán López
- Sergio Del Pino Gómez
- Víctor Pérez Martín

Características del proyecto:

Producto solicitado:

Realización de un proyecto de desarrollo software siguiendo los métodos y técnicas de ingeniería de software vistas en teoría, así como aplicando las herramientas presentadas en las diferentes sesiones de prácticas. El proyecto se basa en la construcción de un sistema informático para la gestión de una biblioteca.

Objetivos del Proyecto:

Antes de abordar este proyecto, se establecieron los siguientes cinco objetivos:

Desarrollar un entorno de trabajo independiente de la metodología de desarrollo.

- Uso de herramientas colaborativas como Office 365 para la comunicación, control y colaboración entre los miembros del equipo.
- Utilización del lenguaje Java para el desarrollo del código, revisando principios básicos de la Programación Orientada a Objetos y aspectos avanzados como patrones de diseño.

Poner en valor una metodología iterativa e incremental, usando versiones y gestión de configuración para planificar el desarrollo del software.

- GitHub como repositorio del proyecto, facilitando la administración de versiones y modificaciones del código.
- Empleo de la herramienta de construcción de software Maven.

Comprender el concepto de calidad del producto de software.

- Identificación de requisitos, tanto funcionales como no funcionales, derivados del quality by design y su incorporación al diseño.
- Resolución de problemas relacionados con la calidad del software.

Estructura del Proyecto:

Se utilizó SCRUM, una metodología de desarrollo ágil basada en un proceso iterativo e incremental, diseñada para ofrecer valor al cliente durante todo el desarrollo del proyecto.

Cronograma de Actividades:

En una metodología de SCRUM, contamos con los siguientes eventos:

- Sprint: Unidad básica de trabajo del equipo.
- Planificación del Sprint: Reunión al inicio de cada Sprint para definir qué realizar y cómo efectuarlo.
- Scrum Semanal: Evaluación del progreso y la tendencia, sincronizando actividades y creando un plan para los próximos 7 días.
- Revisión del Sprint: Muestra el trabajo completado respecto al backlog del producto para futuras entregas.

Para la planificación y gestión de los Sprint, se utilizó la herramienta integrada de GitHub Project.

¿Cómo nos hemos organizado?

Hemos llevado a cabo nuestro proyecto utilizando la metodología de desarrollo ágil Scrum. Durante el proceso, hemos realizado sprints con una duración de una semana cada uno, y hemos programado reuniones semanales los miércoles para acordar el número de tareas a abordar y asignar responsabilidades.

El sprint uno tuvo una duración de dos semanas para organizar y aprender sobre los conceptos de Scrum, además de realizar la preparación del entorno de trabajo y herramientas que emplearíamos durante todo el proyecto.

Los demás sprints han sido de una semana y han ido perfectamente planificados y los tiempos se han cumplido en todos, salvo algunos contratiempos que hemos tenido, en el que hemos superado los tiempos estimados de algunas tareas.

Toda nuestra gestión y organización se ha llevado a cabo en la plataforma GitHub. En esta plataforma, mantenemos un Product Backlog que lista todas las tareas pendientes. Cada tarea en el Product Backlog tiene asignada una prioridad dentro del sprint, una estimación de tiempo en horas, así como fechas de inicio y finalización.

Para el desarrollo del código, hemos utilizado el entorno de desarrollo IntelliJ IDEA, que gracias a sus funciones integradas ha facilitado en gran medida la organización y la sincronización con GitHub. Además, hemos aprovechado la funcionalidad "Code With Me" de IntelliJ IDEA, que nos ha permitido realizar sesiones de trabajo conjunto, modificando el código en tiempo real de manera simultánea entre varias personas que participan en la sesión.

En cuanto al desarrollo técnico, hemos integrado Maven en nuestro código para la verificación de pruebas. También hemos utilizado tecnologías como Spring Boot y Hibernate. En la gestión de la base de datos, hemos empleado JPA (Java Persistence API) y sus anotaciones para facilitar la interacción con la base de datos y la generación de esquemas correspondientes. Este enfoque nos ha proporcionado una estructura robusta y eficiente para la gestión de datos en nuestro proyecto.

Entregables

Para cada entregable, hemos subido el código implementado hasta la fecha en la rama main, que solo tendrá commits por cada entrega del proyecto

Gestión de ramas

Para el desarrollo del software hemos establecido un flujo de trabajo Gitflow, el cual contiene las siguientes ramas:

- La rama *main* o principal almacena el historial de publicación oficial.
- La rama de desarrollo *develop* es creada a partir de la rama main. Sirve como rama de integración para las funciones.
- La rama de lanzamiento *release* se crea a partir de la rama develop.
- Las diferentes ramas *features*, referidas a las diferentes características del software (fGestiónUsuarios, f-GestiónTitulos,…), se crean a partir de la rama develop.
- Cuando una característica *feature* se ha completado, unimos la rama *feature* a la rama desarrollo *develop.*
- Cuando la rama de lanzamiento *release* está hecha, se une a la de desarrollo *develop* y a la principal *main*.
- La rama Hotfix surgió como una rama que permitía realizar cambios urgentes en el código sin afectar a la rama principal directamente.
- La rama de Mantenimiento fue creada principalmente para arreglar los diferentes problemas de mantenimiento y code smells.
- La rama de Testing tiene el objetivo de implementar aquellos métodos para testear las distintas clases y funciones del programa.

Gestión de la base de datos

Elección de MySQL y Justificación:

En el desarrollo de nuestro proyecto, decidimos utilizar MySQL como sistema de gestión de bases de datos (DBMS). La elección de MySQL se basó en la familiaridad que la mayoría de los desarrolladores del equipo tienen con este sistema. MySQL es conocido por ser robusto, escalable y ampliamente utilizado en la industria, lo que facilita el manejo y la colaboración entre los miembros del equipo. Este enfoque proporciona una sólida base para el desarrollo y el crecimiento continuo de nuestro proyecto.

Integración con Google Cloud:

Para alojar nuestra base de datos, optamos por la plataforma de Google Cloud. Google Cloud proporciona un entorno seguro y confiable para alojar bases de datos MySQL, ofreciendo servicios como Cloud SQL, que es un servicio totalmente administrado que facilita la implementación, el mantenimiento y la administración de bases de datos MySQL en la nube.

Configuración de la Conexión:

Para establecer la conexión entre la aplicación y la base de datos MySQL alojada en Google Cloud, se realizó una configuración adecuada. Esto incluyó la obtención de credenciales seguras, como nombres de usuario y contraseñas, así como la especificación de la dirección y el puerto de la base de datos en la nube.

Estructura de la Base de Datos:

La estructura de la base de datos se diseñó cuidadosamente para satisfacer los requisitos específicos del proyecto. Esto implicó definir (con la ayuda de JPA) tablas, campos, relaciones y restricciones para garantizar la integridad y la eficiencia de los datos almacenados. Se realizaron esfuerzos para normalizar la base de datos y evitar redundancias, lo que contribuye a un diseño más eficiente y mantenible.

Uso de MySQL Workbench:

Para facilitar el desarrollo, monitoreo y prueba de la base de datos MySQL alojada en Google Cloud, los desarrolladores y testers utilizaron MySQL Workbench. MySQL Workbench es una herramienta gráfica que proporciona una interfaz intuitiva para administrar y visualizar bases de datos MySQL.

Durante la fase de desarrollo, los desarrolladores utilizaron MySQL Workbench para diseñar visualmente la estructura de la base de datos. Esta herramienta permitió la creación y modificación de tablas, así como la definición de relaciones entre ellas, proporcionando una visión clara y comprensible de la arquitectura de la base de datos.

Durante la fase de pruebas, los testers utilizaron MySQL Workbench para insertar datos de prueba en la base de datos y verificar la integridad de las operaciones de lectura y escritura. La capacidad de visualizar los datos de manera gráfica facilitó la identificación de posibles problemas y errores en el proceso de desarrollo.

Durante el ciclo de vida del proyecto, los desarrolladores y el equipo de pruebas utilizaron MySQL Workbench para monitorear en tiempo real la actividad de la base de datos. Esto incluyó la supervisión de conexiones, el rendimiento de las consultas y otros indicadores clave que ayudaron a garantizar un funcionamiento eficiente del sistema.

Pruebas:

Para la gestión de los diferentes aspectos del testing y de los casos de pruebas, se ha diseñado un plan, cuyo documento se encuentra en la carpeta “PANTYSMEL/planificaciones” dentro del repositorio del proyecto. Estos casos de pruebas se han implementado mediante el uso de Junit y Mockito. Además, se ha automatizado la generación de informes con el uso de Maven. Para acceder al Plan de prueba y casos de pruebas, haz click sobre este link.

Mantenimiento:

En cuanto al mantenimiento, se ha elaborado un plan de Mantenimiento SonarCloud", que puede encontrarse en la carpeta de planificaciones en GitHub. Este plan se ha seguido para el mantenimiento del sistema a través del programa SonarCloud. También se puede encontrar en la carpeta "PANTYSMEL/planificaciones" dentro del repositorio del proyecto.

Identificación de requisitos:

Los requisitos funcionales y no funcionales son dos categorías principales de especificaciones que definen las características y características de un sistema o software. A continuación todos los requisitos que optamos por implementar:

Requisitos Funcionales:

- **Definición:** Estos requisitos describen las funciones específicas que debe realizar el sistema, es decir, las acciones o servicios que el sistema debe proporcionar.
- **Requisitos:**
  - **Crear un AboutUs:** Este es un requisito funcional ya que se refiere a la función específica de crear una página "Acerca de nosotros" en la web.
  - **Crear un ContactUs:** También es un requisito funcional, ya que se trata de la función de crear una página de contacto.
  - **Crear un OurTeam:** Igualmente, este es un requisito funcional, ya que implica la creación de una página específica, en este caso, sobre el equipo.
  - **Crear una forma de registrarse:** Es un requisito funcional porque se refiere a la función de proporcionar a los usuarios la capacidad de registrarse en la página web.
  - **Poder añadir una foto a los títulos creados**: Este requisito es funcional, ya que se refiere a la capacidad específica de añadir imágenes a los títulos de las páginas creadas.

Requisitos No Funcionales:

- **Definición:** Estos requisitos se refieren a las características que no están relacionadas directamente con las funciones específicas del sistema, sino más bien con cómo se deben realizar esas funciones. Estos aspectos a menudo se centran en la calidad del sistema y en cómo se desempeña en ciertos contextos o condiciones.
- **Requisitos:**
  - **Encriptación de contraseñas:** Este es un requisito no funcional, ya que se centra en cómo se deben realizar ciertas funciones (en este caso, la forma en que se almacenan y manejan las contraseñas).
  - **Creación de un icono de la página en función del rol (admin, user):** Este también es un requisito no funcional, ya que se refiere a la presentación visual de la página web basada en roles específicos de administrador y usuario.

Conclusión:

Por último y ya una vez finalizado el proyecto, podemos observar que a lo largo del tiempo de desarrollo se han ido cumpliendo los diferentes objetivos marcados para el mismo, algunos de ellos como la implementación de una metodología ágil de modelo scrum, llevar a cabo un plan de mantenimiento y de prueba para mejorar la calidad del software, hacer un correcto uso de Maven junto a sus dependencias, etc.

Entre todos los conocimientos adquiridos, destacan:

- Haber conocido el alcance global de un producto software.
- Ser capaces de enfocar un proyecto por etapas y objetivos bien marcados.
- Obtener comprensión sobre el funcionamiento de GitHub, SonarCloud, entre otros y cómo estos se aplican al ámbito de la empresa.
- Haber sido capaces de distribuir las diversas áreas de trabajo entre los diferentes roles dentro de la organización.
- Trabajar en equipo de forma eficaz, eficiente y organizada.