MEMORIA – ENTREGA 1

CLAUDIA MARTÍN – ANAIS CHIQUITO – DAVID MARCOS – MIGUEL AHIJÓN

INGENIERIA DEL SOFTWARE II.

CONTENIDO

1. INTRODUCCIÓN

El documento recoge la memoria intermedia del proyecto “HomygoS.L”, desarrollado en la asignatura de Ingeniería del Software II.

El objetivo principal del proyecto es construir una aplicación web para la gestión de alquileres de viviendas entre particulares, similar en conceptos a plataformas como Airbnb, aplicando metodologías agiles y herramientas profesionales de desarrollo colaborativo.

El sistema permite que propietarios registren inmuebles o habitaciones disponibles y que usuarios puedan buscar, reservar y gestionar sus alquileres a través de un entorno web. Además, se contemplan diferentes modos de reserva (inmediata o bajo solicitud), así como el uso de filtros avanzados para mejorar la experiencia de búsqueda.

Esta memoria tiene como propósito presentar los avances logrados durante los primeros sprints, detallando la planificación, metodología empleada, herramientas, estructura técnica del sistema y la distribución de tareas dentro del equipo.

2. PLANIFICACIÓN Y GESTIÓN

2.1. METODOLOGÍA DE DESARROLLO (SCRUM)

El equipo adopto Scrum como metodología principal, organizando el desarrollo en sprints cortos y planificados, con revisiones periódicas.

Las fases de Scrum aplicadas fueron:

1. Sprint Planning: selección de historias de usuario y definición de tareas.
2. Daily Scrum: coordinación diaria del equipo.
3. Sprint Review: evaluación de resultados parciales.
4. Retrospective: análisis de eficiencia y mejoras de proceso.

2.2. ROLES DEL EQUIPO

El equipo cuenta con cuatro integrantes, distribuidos en los siguientes roles:

Rol	Responsabilidad principal
Product Owner	Definir y priorizar el backlog, validar entregas
Srum Master	Asegurar la correcta aplicación de Scrum y eliminar bloqueos
Desarrollador Backend	Implementar la lógica del sistema, entidades y persistencia
Desarrollador Frontend	Desarrollar vistas, controladores y diseño web

2.3. HERRAMIENTAS DE PLANIFICACIÓN

- GitHub Projects: gestión del backlog y tareas mediante tablero Kanban (To Do, In Progress, Done)
- GitHub Milestones: control de hitos y seguimiento por sprints
- Google Docs: documentación
- Maven: soporte de gestión de dependencias y ejecución

2.4. PLANIFICACIÓN POR SPRINTS

El trabajo se estructuro en tres Sprints principales, registrados como milestones en GitHub:

Sprint 1 – Inicio del proyecto

- Configuración del entorno de trabajo con Spring Boot.
- Creación del proyecto y configuración del pom.xml.
- Implementación del registro e inicio de sesión básicos.
- Primera conexión con la base de datos (Spring Data JPA).
- Comprobación de funcionamiento de persistencia.

Sprint 2 – Desarrollo del modulo de recursos

- Configuración de src/main/resources y del archivo application.properties.
- Creación de plantillas HTML (login, registro, inmuebles).
- Desarrollo de entidades Usuario, Propietario y Vivienda.
- Implementación de controladores y flujo MVC.
- Revisión de vistas y estructura básica del sitio.

Sprint 3 – Mejora de interfaz y controladores (en curso)

- Optimización de vistas HTML y mejora de usabilidad.
- Creación de nuevas clases MVC y refactorización de controladores.
- Ampliación de la capa de servicios y conexión entre módulos.

2.5. DISTRIBUCIÓN DE TAREAS POR INTEGRANTE

Integrante	Rol	Funciones
Miguel Ahijón	Product Owner	Definición de backlog, validación de historias de usuario y revisión de requisitos
Claudia Martín	Scrum Master	Coordinación del equipo, control de entregas y soporte técnico
David Marcos	Backend Developer	Programación de la lógica de negocio, servicios y conexión con la base de datos
Anaís Chiquito	Frontend Developer	Diseño y desarrollo de vistas HTML y comunicación MC

3. CODIFICACIÓN

3.1. TECNOLOGÍAS Y ARQUITECTURA

El proyecto se implementó con Spring Boot 3 bajo el lenguaje Java, utilizando el patrón Modelo-Vista-Controlador (MVC), garantizando modularidad y mantenibilidad del código.

Tecnologías principales:

- Spring Boot: framework base para la aplicación.
- Spring Data JPA: acceso a base de datos MySQL.
- Thymeleaf: motor de plantillas para vistas dinámicas.
- Maven: gestor de dependencias y compilación.

3.2. COMPONENTES PRINCIPALES

- Modelos: Usuario, Propietario, Vivienda, Reserva.
- Servicios: Gestión de reservas, persistencia y validaciones.
- Controladores: Coordinación entre las vistas y la capa de negocio.
- Vistas: HTML + CSS con Thymeleaf, para login, registro y gestión de propiedades.

3.3. AVANCES TÉCNICOS

Durante la entrega intermedia se logró:

- Crear una arquitectura base estable y funcional.
- Implementar la comunicación entre capas MVC.
- Desarrollar la persistencia con MySQL.
- Construir una interfaz funcional de usuario.

Se siguieron buenas prácticas de programación:

- Inyección de dependencias con @Autowired.
- Comentado y documentación del código.
- Manejo de errores y validaciones básicas.
- Convenciones de nombres consistentes.

4. GESTIÓN DE CONFIGURACIÓN

4.1. CONTROL DE VERSIONES

El proyecto se gestiona mediante Git y GitHub, con el flujo de trabajo basado en ramas:

- main: rama estable para entregas
- develop: integración de funcionalidades en curso
- feature/*: ramas especificas por historia de usuario o tarea

Cada commit describe los cambios realizados, y los pull requests garantizan la revisión entre integrantes antes de la integración.

4.2. GESTIÓN DE DEPENDENCIAS Y COMPILACIÓN

El archivo pom.xml contiene las dependencias necesarias del proyecto:

library > pom.xml

```xml
<project xmlns="http://maven.apache.org/POM/4.0.0"
    <dependencies>
        <!-- Dependencias básicas -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-thymeleaf</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-tomcat</artifactId>
            <scope>provided</scope>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>

        <!-- Dependencias para Spring JPA y Derby -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-jpa</artifactId>
        </dependency>
        <dependency>
```

El uso de Maven facilita la construcción del proyecto, ejecución de pruebas unitarias y generación del artefacto .jar.

4.3. INTEGRACIÓN Y DESPLIEGUE

- Servidor: Tomcat embebido en Spring Boot.
- Base de datos: MySQL local configurada en application.properties.
- Pruebas: Validación del login, registro y persistencia de entidades.

Se planifica la integración de pruebas automáticas y la futura implementación de un entorno de despliegue remoto para la entrega final.

5. CONCLUSIÓN

Durante la entrega intermedia, el proyecto Homygo S.L. ha cumplido con los objetivos establecidos en las tres áreas evaluables del contrato:

- Planificación y gestión: trabajo iterativo siguiendo Scrum, con planificación real en GitHub y roles definidos.
- Codificación: estructura sólida bajo arquitectura MVC con componentes funcionales.
- Gestión de configuración: uso eficiente de Git, ramas, Maven y control de dependencias.

El equipo mantiene una colaboración activa y ha logrado construir una base sólida para la entrega final, donde se completarán las funcionalidades de reservas, pagos y optimización del sistema.