Galaxy Impact V – Análisis de requisites

Nombre del proyecto: Galaxy Impact V

Descripción breve: Galaxy Impact V es un shooter donde el jugador enfrenta oleadas de invasores extraterrestres. El juego se desarrolla con Unity (cliente) y se conecta a una API REST en Java (Spring Boot) con una base de datos MySQL para la gestión de usuarios, leaderboard y guardado de progreso.

1. Objetivos principales

Objetivo general

Desarrollar un shooter con una jugabilidad atractiva y un sistema de persistencia mediante una API propia en Java y una base de datos MySQL.

Objetivos específicos

1. Implementar el motor de juego en Unity con movimiento, disparo y oleadas de enemigos.
2. Crear una API REST en Java (Spring Boot) para autenticación, almacenamiento de puntuaciones (leaderboard) y guardado del progreso del jugador.
3. Almacenar datos en MySQL: usuarios, puntuaciones y progreso.
4. Documentar la arquitectura, la API (OpenAPI/Swagger) y procedimientos para desplegar el proyecto.

2. Perfil de usuario (persona objetivo)

Cualquier persona mayor de 12 años que quiera jugar a un juego simple y entretenido.

3. Funcionalidades previstas

- Movimiento del jugador en 8 direcciones.
- Disparo apuntando al cursor.
- Enemigos alienígenas de varios tipos: rápidos, resistentes y voladores.
- Modos de juego: Supervivencia y modo historia
- HUD: vida, munición, puntuación y oleada actual o misión actual.

- Pantallas: Menú principal, pausa y game over.
- Efectos visuales: Animaciones del personaje, impactos, explosiones y partículas.
- Variedad de mapas
- Registro y login de usuarios.
- Guardado de puntuaciones
- Guardado de progreso, desbloqueables y logros.

4. Requisitos funcionales (RF)

Jugabilidad

RF1. El jugador puede moverse en 8 direcciones dentro del mapa.
RF2. El jugador puede apuntar y disparar en la dirección del cursor.
RF3. El sistema genera oleadas de enemigos con dificultad creciente.
RF4. El juego muestra un HUD con vida, munición, puntuación y oleada actual.
RF5. El juego dispone de pantallas de menú, pausa y game over.

Backend / API

RF6. El sistema permite registrar y autenticar usuarios (`POST /api/auth/register`, `POST /api/auth/login`).
RF7. El sistema acepta el envío de puntuaciones (`POST /api/scores`) y las almacena en MySQL.
RF8. El sistema proporciona el leaderboard público (`GET /api/leaderboard`).
RF9. El cliente Unity puede recuperar datos de usuario y leaderboard mediante peticiones HTTP.

5. Requisitos no funcionales (RNF)

RNF1 (Rendimiento). El juego se mantendrá fluido y estable en un PC estándar durante las oleadas del PMV.
RNF2 (Latencia API). Las respuestas de la API para leaderboard y guardado deberán llegar en menos de 1 s en condiciones normales de red.
RNF3 (Usabilidad). Controles intuitivos y HUD claro; feedback visual inmediato al impactar o recoger power-ups.
RNF4 (Mantenibilidad). Código modular y documentado (tanto en Unity como en el backend) con control de versiones.

6. Tecnologías previstas

- **Cliente (juego):** Unity (C#)
- **Backend / API:** Java con Spring Boot.
- **Base de datos:** MySQL
- **Comunicación:** REST JSON; Unity utiliza `UnityWebRequest`.
- **Assets:** Blender para gráficos 2D si se crean recursos propios.

7. Producto mínimo viable (PMV)

1. Movimiento y disparo del jugador.
2. Enemigos básicos y oleadas progresivas (mínimo 3).
3. HUD y pantallas básicas (menú, pausa, game over).
4. Registro/login de usuario.
5. Guardado de puntuación al finalizar la partida y leaderboard público.
6. Base de datos MySQL con `users` y `scores`.