Institut Puig Castellar
Projecte / Crèdit de síntesi
Document funcional

DOCUMENTO FUNCIONAL DEL PROYECTO

ALUMNOS/GRUPO: Adrián Luque y Justin Alvarez

---

1. Introducción y contexto

- **El problema o necesidad que se quiere resolver:** Existe una carencia de plataformas interactivas que permitan practicar conceptos de lógica computacional, algoritmia y programación de una forma lúdica y gamificada, alejándose de los entornos de estudio tradicionales.
- **Quién será el usuario o cliente final:** Estudiantes de ciclos formativos de informática, desarrolladores junior y aficionados a la programación que busquen poner a prueba sus conocimientos.
- **Qué solución se propone y con qué propósito:** Se desarrollará "BitHub", una plataforma web arcade que ofrecerá un catálogo de minijuegos técnicos (binario, lógica booleana, sintaxis, etc.). Su propósito es fomentar el aprendizaje mediante un sistema de economía virtual (Bytes) y clasificaciones globales que estimulen la competitividad sana.

2. Análisis de requisitos

2.1. Requisitos funcionales (RF)

Qué debe hacer el sistema. Enumera las funciones principales.

Código	Descripción del requisito funcional
RF1	El sistema permitirá a los visitantes registrarse e iniciar sesión de forma segura.
RF2	El sistema asignará un saldo virtual inicial (Bytes) a cada nuevo usuario.

RF3	El sistema permitirá al usuario acceder y jugar a diversos minijuegos, descontando el coste de entrada y sumando las recompensas en caso de victoria.
RF4	El sistema generará una clasificación global (ranking) basada en el saldo de los usuarios.
RF5	El sistema dispondrá de un panel de administración para gestionar el contenido de los minijuegos y visualizar estadísticas de uso.

2.2. Requisitos no funcionales (RNF)

Cómo debe comportarse el sistema.

Código	**Descripción del requisito no funcional**
RNF1	La interfaz web deberá ser completamente adaptativa (*responsive*) para dispositivos móviles y de escritorio.
RNF2	El sistema deberá encriptar las contraseñas de los usuarios en la base de datos (mediante BCrypt).
RNF3	Las transacciones de saldo (Bytes) deben procesarse en tiempo real sin requerir la recarga manual de la página.
RNF4	La plataforma deberá alcanzar métricas de rendimiento superiores a 90/100 en auditorías de Lighthouse (LCP < 2s).

2.3. Restricciones

Condiciones o limitaciones del proyecto.

- **Lenguajes o tecnologías obligatorias:** Vue.js (TypeScript) para el cliente, Java (Spring Boot) para el servidor y MySQL para la persistencia de datos.
- **Recursos disponibles:** Tiempo limitado al calendario académico del proyecto. Uso exclusivo de herramientas y plataformas de despliegue de capa gratuita (*Free Tier*).

- **Dependencias o limitaciones técnicas:** La comunicación entre cliente y servidor dependerá estrictamente del correcto despliegue de la API REST en la nube y de la disponibilidad del proveedor de base de datos.

3. Análisis de usuarios y roles

Identificar quién usará el sistema y qué podrá hacer.

Rol	Descripción	Permisos principales
Administrador	Encargado del mantenimiento de la plataforma y contenido.	Acceso al panel de administración, modificación de preguntas/cartas, visualización de métricas globales.
Usuario (Jugador)	Usuario registrado que interactúa con la plataforma.	Jugar minijuegos, gestionar su saldo (Bytes), consultar la clasificación global.
Visitante	Usuario sin autenticar.	Acceso exclusivo a la pasarela de registro y autenticación. Navegación bloqueada.

4. Casos de uso / Escenarios de uso

Cómo interactúan los usuarios con el sistema.

Código	Nombre del caso de uso	Actor principal	Descripción	Resultado esperado
CU1	Autenticación de usuario	Visitante	El usuario introduce sus credenciales en el formulario de inicio de sesión.	Se valida la identidad y se redirige al catálogo de juegos (Home).
CU2	Participación en minijuego	Usuario	El usuario selecciona un juego, el sistema resta la entrada de	La partida finaliza y se actualiza el saldo según el resultado (victoria/derrota).

			su saldo e inicia la partida.	
CU3	Consulta de clasificación	Usuario	El usuario accede a la vista de ranking.	Se muestra una lista ordenada de jugadores según su cantidad de Bytes.
CU4	Gestión de contenido	Administrador	Accede al panel de control para añadir una nueva pregunta al juego Logic Slots.	La base de datos se actualiza y la nueva pregunta está disponible en el juego.

5. Modelo de datos o estructura de la información

Representación de la información que gestionará el sistema.

- **Entidades principales:**
  - Usuarios: Almacena credenciales, rol (ADMIN/USER) y saldo actual (Bytes).
  - Partidas_Stats: Registra el historial de juego (Juego, resultado, cantidad ganada/perdida, fecha).
  - Preguntas / Cartas: Almacena el contenido dinámico necesario para la ejecución de los minijuegos.
- **Relaciones:**
  - Un *Usuario* puede registrar múltiples *Partidas* (Relación 1:N).
  - Los minijuegos consumen aleatoriamente registros de las tablas de *Preguntas/Cartas*.

6. Diseño de la interfaz

Estructura y navegación del sistema.

- **Vista de Login/Registro:** Funcionalidad de autenticación. Interfaz minimalista para el acceso al sistema (relacionado con CU1).
- **Home (Salón Arcade):** Funcionalidad principal. Muestra el perfil, saldo en tiempo real y el catálogo de juegos disponibles (relacionado con CU2).
- **Vistas de Minijuegos (DeciBit, Code-Jack, etc.):** Tableros de juego interactivos donde se desarrolla la lógica de cada reto.
- **Clasificación (Ranking):** Tabla dinámica que expone las puntuaciones globales (relacionado con CU3).
- **Panel de Administración:** Menú lateral de gestión (CRUD) de preguntas, cartas y visualización de registros estadísticos (relacionado con CU4).

7. Planificación técnica

Tecnologías y herramientas a utilizar.

- **Lenguajes y frameworks:** Vue.js (3) con TypeScript para el Frontend; Java 17 con Spring Boot para el Backend.
- **Base de datos:** MySQL (Desplegada en la nube).
- **Herramientas de diseño o edición:** Figma para prototipado, Visual Studio Code / IntelliJ IDEA para desarrollo.
- **Despliegue:** Vercel (Frontend) y Railway (Backend y Base de datos). Git/GitHub para control de versiones.
- **Reparto de tareas:** Adrián Luque y Justin Alvarez (Desarrollo integral Full-Stack, repartiendo la lógica de componentes de Vue y controladores de Spring equitativamente).

8. Análisis de riesgos

8.1. Identificación de riesgos

- **R1:** Dificultad técnica o curva de aprendizaje pronunciada en la integración Vue - Spring Boot.
- **R2:** Errores de sincronización de datos o pérdida de conexiones en los despliegues gratuitos.
- **R3:** Vulnerabilidades en la manipulación del saldo (Bytes) desde el lado del cliente.
- **R4:** Falta de tiempo para finalizar todos los minijuegos propuestos.

8.2. Valoración y respuesta

Riesgo	Probabilidad	Impacto	Plan de prevención o contingencia
R1	Media	Alta	Desarrollar pruebas de concepto tempranas (PoC) y realizar tutorías/revisiones periódicas.
R2	Baja	Alta	Mantener el código versionado en GitHub; tener bases de datos de respaldo en local.
R3	Media	Alta	Mover toda la lógica matemática y de validación de recompensas al Backend.
R4	Alta	Media	Priorizar el motor principal y 2 juegos básicos; dejar los demás como objetivos secundarios.

9. Validación y criterios de éxito

- **Criterios de aceptación:** El sistema permite el ciclo completo de un jugador: registro, ganancia/pérdida de Bytes mediante el juego, y visualización correcta en el ranking.

- **Pruebas previstas:** Pruebas funcionales de interfaz (botones, rutas protegidas) y auditorías de rendimiento automatizadas mediante Google Lighthouse.
- **Indicadores de calidad:** Lograr cero errores en consola durante la ejecución y alcanzar la máxima puntuación (100/100) en el apartado de rendimiento y buenas prácticas.

10. Conclusión

Durante la fase de análisis se ha determinado que "BitHub" constará de una arquitectura robusta Full-Stack orientada a ofrecer un servicio gamificado, seguro y de alta disponibilidad. Se ha optado por tecnologías líderes (Vue y Spring Boot) que garantizan agilidad y escalabilidad.

Próximos pasos:

1. Preparar el entorno de desarrollo y clonar los repositorios iniciales.
2. Crear la estructura relacional de la base de datos en MySQL y conectar la API REST.
3. Empezar la implementación del sistema de autenticación (CU1) y la interfaz principal (Home).

El análisis funcional proporciona una visión clara y realista del proyecto. A partir de aquí, comienza la fase de desarrollo para materializar las mecánicas de BitHub.