ANTEPROYECTO "DAM"

Índice

---

1. TÍTULO

ZENHABITS: Aplicación móvil para la gestión de hábitos, tareas y metas con "Gamificaión"

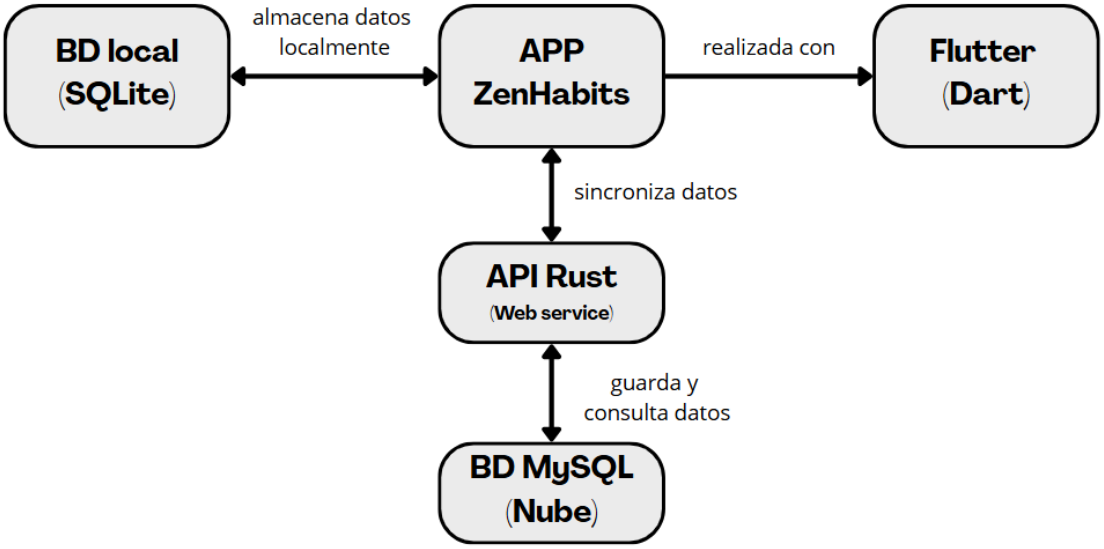
2. DEFINICIÓN DEL PROBLEMA

Actualmente, las personas quieren conseguir mantener hábitos saludables y ser productivos. Hay aplicaciones que colaboran con ello, pero pocas son efectivas. Por ello propongo:

- ZENHABITS: una aplicación diseñada para ayudar a los usuarios a gestionar sus hábitos y metas diarias mediante un sistema de recompensas: Se podrán obtener logros y habrá un personaje el cual podrá subir de nivel o personalizarlo. Esto se podrá en base al cumplimiento de hábitos, tareas diarias y metas (si por ejemplo no cumples las tareas diarias, el personaje perderá vida) Además, se podrán añadir desafíos para incentivar la constancia.

Se desarrollará con Flutter para garantizar compatibilidad multiplataforma (Android/iOS) y contará con una API en Rust para la gestión de datos.

3. ESQUEMA O GRÁFICO DEL PLANTEAMIENTO



La aplicación se desarrollará con Flutter (Dart), esta guardará los datos de manera local (SQLite) y sincronizará estos datos con una API desarrollada en Rust. La API actuará como "intermediaria", guardando y consultando información en una base de datos en la nube (MySQL). De esta forma, garantizaré el uso de los datos con o sin conexión.

- **Análisis de requisitos del usuario:**

Para que la aplicación sea efectiva y atractiva, existen varios requisitos que debe cumplir; algunos de ellos serían:

  - Gestión de hábitos, tareas diarias y/o metas.
  - Crear, modificar y eliminar hábitos, tareas diarias y/o metas.
  - Configurar notificaciones para los recordatorios.
  - Almacenamiento de datos local para uso sin conexión.
  - Integración con la API en Rust para sincronización con el servidor.

- **Análisis del problema:**

Actualmente, las aplicaciones de gestión de tareas, hábitos y organización; presentan dos problemas:

➔ Falta de aplicaciones que ayuden al usuario a ser constante.

➔ Complejidad en la sincronización de datos entre plataformas (dependen completamente de la nube obligando a tener conexión).

- **SOLUCIÓN:**

  App móvil "gamificada": está tendrá un sistema de "recompensas" por cumplir sus objetivos y desafíos donde los usuarios podrán desbloquear logros.

  Sincronización Híbrida:

  - SQLite: El almacenamiento local con SQLite permitirá que los usuarios puedan usar las funciones de la aplicaión sin conexión a internet.
  - Sincronización en la Nube: Cuando el dispositivo esté conectado a internet, los datos se podrán sincronizar con la API en rust, asegurando la disponibilidad y accesibilidad en múltiples dispositivos.

- **Diseño:**

  La aplicación móvil será desarrollada con Dart en Visual Studio Code o Android Studio *(está por decidir)*, utilizando Flutter para que sea multiplataforma y tenga una interfaz atractiva y funcional. Haré un diseño limpio y minimalista.

  ➔ ESTRUCTURA DE LA APLICACIÓN:
  - Pantalla de bienvenida y registro/inicio de sesión del usuario.
  - Panel principal (inicio) con lista de hábitos y personaje (menú inferior para navegar entre hábitos, tareas diarias y metas).
  - Formularios para la creación de hábitos, tareas y metas.
  - Posible pantalla de configuración de notificaciones, logros...

  ➔ PROTOTIPOS: Para garantizar un buen diseño antes de implementar nada haré el diseño de los prototipos (Canva) y diseño de diagramas de navegación y distribución de elementos en cada pantalla y realizaré pruebas de usabilidad con usuarios reales para validar la facilidad de uso.

- **Pruebas:**
  - Pruebas unitarias.
  - Pruebas de carga y rendimiento en la API.
  - Se utilizarán emuladores para verificar la compatibilidad con diferentes modelos y plataformas y también en dispositivos físicos reales.

➔ TESTING DE INTERFAZ DE USUARIO (UI/UX)

- Test de usabilidad: Se realizan pruebas con usuarios reales para identificar posibles puntos de fricción. Esto ayuda a ajustar el diseño y a mejorar la experiencia de usuario.

4. OBJETIVOS

Desarrollar una aplicación móvil que permita la creación y seguimiento de hábitos, tareas diarias y metas mediante un sistema "gamificado" para mantener la constancia de los usuarios, además, de forma personal, aprender nuevos lenguajes y herramientas para desarrollar aplicaciones.

Objetivos específicos:

- Implementar una interfaz amigable y funcional en Flutter para posibilidad multiplataforma.
- Desarrollar una API segura en Rust para la gestión de usuarios y datos.
- Asegurar la sincronización de datos en la nube para acceso desde varios dispositivos.
- Integrar un sistema de recompensas y notificación.

5. PROCEDIMIENTO

- **Entornos de desarrollo:**
  La aplicación se desarrollará en Visual Studio Code o Android Studio (está por decidir, pero priorizaré la opción más sencilla y flexible para el desarrollo en Flutter).
- **Bases de datos utilizadas:**
  SQLite para la base de datos local y MySQL en el servidor para la sincronización de datos.
- **Lenguajes utilizados:**
  - Dart para el desarrollo de la lógica de la aplicación e interfaz de usuario con Flutter.
  - Rust para la creación de la API.
- **Sistemas operativos donde residirá:**
  Android e iOS (multiplataforma gracias a Flutter)

- **Tipo de dispositivos:**
  Principalmente orientada a móviles y tablets (posibilidad de adaptar la interfaz para su uso en escritorio y web en futuras fases del proyecto)
- **Usabilidad de los diseños:**
  Diseños centrados en la experiencia del usuario (UX), con interfaces intuitivas, accesibles y visualmente agradables y minimalistas.
- **Herramientas de virtualización:** Se contempla el uso de Docker para virtualización y despliegue

Actividades:

Análisis del proyecto, diseño de la aplicación, diseño de la interfaz, implementación Dart, desarrollo de la interfaz con Flutter, desarrollo de la API en Rust, integración de la API con la app móvil, pruebas y control de versiones

Descripción de las actividades:

1. Análisis del proyecto: Concretar el objetivo de la aplicación, análisis de requisitos funcionales y no funcionales, evaluar las tecnologías y herramientas a utilizar y crear de un plan de desarrollo.

2. Diseño de la aplicación**:**

   - Realizar diagramas de flujo y diagramas de entidad-relación (ERD) para la base de datos.
   - Estructurar los módulos y funcionalidades de la aplicación.
   - Definir la arquitectura de la aplicación (por ejemplo: Clean Architecture, MVC, MVVM).

3. Diseño de la interfaz: Diseño adaptado para compatibilidad multiplataforma:

   - Uso de herramientas como Canva, Pencil o Figma para prototipado.
   - Diagramas de navegación.

4. Implementación Dart: Desarrollo de la lógica de negocio utilizando Dart para la gestión de hábitos, metas y demás funcionalidades principales además de la gestión de datos local.

5. Desarrollo de la interfaz con Flutter: Uso de widgets predefinidos de Flutter para optimizar el desarrollo e implementación de temas y estilos consiguiendo una buena experiencia de usuario; tanto en Android como en iOS.

6. Desarrollo de la API en Rust: para garantizar alta eficiencia y seguridad. Esta tendrá conexión con la base de datos en MySQL para el almacenamiento y consulta de datos.
    - Creación de endpoints para la comunicación con la aplicación móvil.
    - Uso de frameworks para la implementación de la API (Axum)
7. Integración de la API con la app móvil:
    - Configuración de peticiones HTTP.
    - Implementación de sincronización de datos entre la base de datos local (SQLite) y el servidor mediante la API.
    - Posiblemente, manejo de autenticación y seguridad con JWT (JSON Web Token).
8. Pruebas: Ejecución de pruebas unitarias para verificar la funcionalidad de cada módulo además de asegurar la correcta comunicación entre la API con la aplicación, y pruebas con usuarios para evaluar la experiencia y, si es necesario, mejorarla.
9. Control de versiones: Se mantendrá un control de versiones del proyecto durante todo su desarrollo utilizando Git y GitHub:
    - Uso de Git para el control de versiones del código.
    - Uso de GitHub para almacenamiento y seguimiento de cambios mediante un repositorio.

6. BIBLIOGRAFÍA

- Dart documentación: https://dart.dev/docs
- Flutter documentación: https://docs.flutter.dev/
- Rust documentación: https://doc.rust-lang.org/book/
- Canva: https://www.canva.com/es_es/
- Mi GitHub (aún no está creado el repositorio ni nada): https://github.com/albealvant
- Uso de IAs como ChatGPT.

También requeriré ayuda del profesor especialista.

Esta bibliografía aumentará mucho a lo largo del proyecto.