**Informe de arquitectura del sistema para “FlashMatch”**



**INTEGRANTES:**

* **Carlos Guzmán Oyarzun**
* **Daniel González Chávez**
* **Rodrigo Berrios Castillo**

**PROFESOR:**

* **Fabian Alejandro Alcántara Guajardo**

**CAPSTONE**

PTY4614-006V

DUOC UC SAN BERNARDO

18/11/2024

**ÍNDICE DE CONTENIDO**

[**1. Introducción 3**](#_7j8xp3euar8b)

[**2. Descripción general del sistema. 4**](#_ctp81ey68jfe)

[**3. Arquitectura del Sistema 5**](#_abj5726pgg97)

[3.1. Componentes del Sistema 5](#_4olxv9g3rkt)

[3.1.1. Frontend 5](#_vy1hka8n8046)

[3.1.2. Backend 5](#_3fgvwfw93c1)

[3.1.3. Base de Datos 6](#_yv16ehs9c3q9)

[3.1.4. Inteligencia Artificial 6](#_ikvoyodc5tqe)

[3.1.5. Notificaciones 6](#_hcdzku4eahh1)

[3.1.6. Pagos 7](#_3jihethamwxd)

[3.2. Flujo de Datos 8](#_sedak6c4vhla)

[3.3. Diagrama de Arquitectura 9](#_enhezeguw64c)

[**4. Consideraciones de Implementación 10**](#_iu6m492qxns1)

[4.1. Seguridad 10](#_5mic0iuxse5m)

[4.2. Escalabilidad 10](#_y25i72adu45k)

[4.3. Mantenimiento 10](#_g95fhcfir4zg)

[4.4. Integración Continua y Despliegue 10](#_hormj94alerl)

[**5. Estrategia de Implementación de IA 11**](#_7019yisk8kaj)

[5.1. Recopilación de Datos 11](#_4ntu0gfhirpm)

[5.2. Desarrollo de Modelos de IA 11](#_ttgxrx7o5eom)

[5.3. Integración en el Sistema 11](#_zag7zfcjyhac)

[**6. Conclusión 12**](#_tg9nm7w6281l)

# 1. Introducción

El presente informe detalla la arquitectura del sistema para **"FlashMatch"**, una plataforma diseñada para conectar a jugadores de fútbol y facilitar la búsqueda y reserva de canchas. Esta aplicación móvil, con una presentación en página web no solo permite a los usuarios organizar partidos, sino que también ofrece recomendaciones personalizadas a través de un sistema de inteligencia artificial. Este documento aborda la estructura del sistema, sus componentes, el flujo de datos y las estrategias de implementación, asegurando que se sigan las mejores prácticas de desarrollo de software.

# 

# 2. Descripción general del sistema.

**"FlashMatch"** tiene como propósito crear una comunidad activa de futbolistas que facilite la búsqueda de canchas y la gestión de partidos. Entre las funcionalidades clave de la aplicación se destacan:

* **Interfaz Intuitiva**: Diseñada para ser accesible en dispositivos móviles, garantizando una experiencia de usuario fluida y atractiva.
* **Gestión de Partidos**: Permite a los usuarios encontrar partidos disponibles en su área, o directamente crear uno para que se unan más usuarios a estos según sus parámetros configurados en el perfil.
* **Recomendaciones Personalizadas**: Utilizando un sistema de IA, se generan recomendaciones de partidos y compañeros/rivales basados en distintos parámetros de los usuarios.

# 

# 3. Arquitectura del Sistema

## 3.1. Componentes del Sistema

### 3.1.1. Frontend

* **Aplicación Web (Angular 18)**:
  + **Descripción**: Utiliza Angular 18 para ofrecer una experiencia web dinámica y responsiva. La interfaz permite a los usuarios interactuar con todas las funcionalidades disponibles de manera eficiente.
  + **Estructura**: Se basa en un diseño modular, donde cada componente maneja una parte específica de la funcionalidad, promoviendo la reutilización del código.
  + **Estilos**: Se implementan estilos CSS modernos utilizando Tailwind CSS, asegurando un diseño limpio y coherente.
* **Aplicación Móvil (Ionic Angular)**:
  + **Descripción**: La aplicación móvil está construida con Ionic, permitiendo un desarrollo multiplataforma. Esto asegura que el producto esté disponible tanto en iOS como en Android con una única base de código.
  + **Funcionalidades Nativas**: Integra capacidades como geolocalización y acceso a la cámara, mejorando la experiencia del usuario al permitir subir imágenes y ubicar canchas cercanas.

### 3.1.2. Backend

* **API REST (NestJS)**:
  + **Descripción**: La API se construye usando NestJS, un framework moderno y modular que permite crear aplicaciones escalables y mantenibles. Utiliza TypeScript, lo que mejora la robustez del código.
  + **Autenticación y Autorización**: Implementa autenticación JWT para asegurar que solo los usuarios validados accedan a los recursos. También se definen roles y permisos para controlar el acceso a diferentes funcionalidades de la aplicación.
* **Gestión de Datos (TypeORM)**:
  + **Descripción**: TypeORM actúa como un ORM (Object-Relational Mapping) que facilita la interacción con la base de datos PostgreSQL. Esto permite realizar operaciones CRUD de manera sencilla y eficiente.
  + **Modelado de Datos**: Se definen entidades que representan las tablas de la base de datos, optimizando la gestión y la consulta de datos.

### 3.1.3. Base de Datos

* **PostgreSQL**:
  + **Descripción**: Se utiliza PostgreSQL como sistema de gestión de bases de datos relacional. Su robustez y capacidad para manejar grandes volúmenes de datos lo convierten en una opción ideal para **"FlashMatch"**.
  + **Estructura**: La base de datos incluye tablas para usuarios, canchas, partidos, niveles de habilidad, estadísticas, y registros de actividad.
  + **Optimización**: Se implementan índices y consultas optimizadas para garantizar un rendimiento eficiente en la búsqueda y manipulación de datos.

### 3.1.4. Inteligencia Artificial

* **Microservicios IA (Python)**:
  + **Descripción**: Desarrollados en Python, estos microservicios se encargan de procesar datos y generar recomendaciones utilizando técnicas de machine learning y deep learning.
  + **Entrenamiento de Modelos**: Se utilizarán algoritmos de clasificación y regresión para predecir las preferencias de los usuarios basándose en su historial y comportamiento.
  + **Integración**: Los microservicios se comunicarán con el backend a través de una API REST, permitiendo la consulta de modelos y la obtención de recomendaciones en tiempo real.

### 3.1.5. Notificaciones

* **Firebase Cloud Messaging (FCM)**:
  + **Descripción**: FCM se utiliza para enviar notificaciones push a los dispositivos móviles, lo que mantiene a los usuarios informados sobre eventos relevantes, como partidos creados o cambios en la disponibilidad de canchas.
  + **Personalización**: Se implementan mecanismos para segmentar usuarios y enviar notificaciones personalizadas basadas en su actividad y preferencias.

### 

### 3.1.6. Pagos

* **Mercado Pago**:
  + **Descripción**: es una plataforma de pagos en línea perteneciente a **MercadoLibre**, una de las mayores empresas de comercio electrónico en América Latina. Su propósito principal es facilitar transacciones electrónicas seguras entre compradores y vendedores, tanto en sitios web como en aplicaciones móviles.
  + **Personalización**: Se implementará principalmente para realizar el pago de las reservas y cupos para cada usuario en el partido.

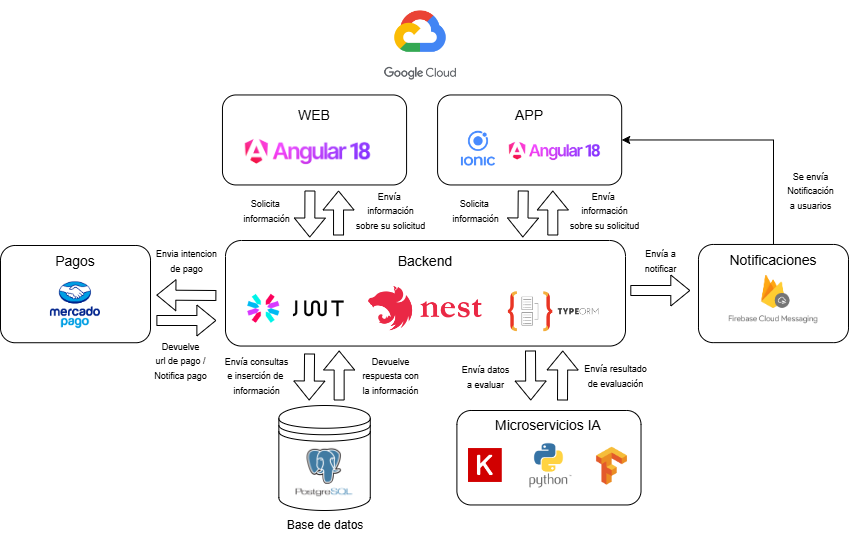
## 3.2. Flujo de Datos

El flujo de datos en **"FlashMatch"** se describe a continuación:

1. **Interacción del Usuario**:
   * Los usuarios inician sesión en la aplicación, buscan equipos o crean partidos a través de la interfaz móvil.
2. **Solicitud a la API REST**:
   * Las acciones del usuario generan solicitudes HTTP a la API REST, que valida la autenticación y autorización del usuario antes de procesar la solicitud.
3. **Gestión de Datos**:
   * La API REST interactúa con la base de datos PostgreSQL a través de TypeORM, recuperando, insertando o modificando datos según sea necesario.
   * Los resultados se devuelven al frontend en formato JSON.
4. **Inteligencia Artificial**:
   * Al solicitar recomendaciones, el backend envía datos relevantes a los microservicios de IA.
   * Estos microservicios procesan la información, ejecutan modelos de machine learning y devuelven recomendaciones optimizadas al backend.
5. **Notificaciones Push**:
   * Cuando ocurre un evento relevante, el backend envía una notificación a FCM, que se encarga de entregar la alerta a los dispositivos móviles de los usuarios.
6. **Pagos:**
   * Cuando se requiere realizar un pago, se envía la solicitud a Mercado Pago para que este devuelva una url para redirigir al usuario a su plataforma de pago.
   * Luego de efectuar el pago por parte del usuario, Mercado Pago envía una notificación de pago al backend para avisar que este fué efectuado con éxito.

### 

## 3.3. Diagrama de Arquitectura



## 

# 4. Consideraciones de Implementación

## 4.1. Seguridad

* **Autenticación y Autorización**:
  + Se implementará un sistema de autenticación sólido que utiliza JWT para manejar sesiones de usuario.
  + Roles y permisos se definirán a nivel de API para controlar el acceso a las funcionalidades según el tipo de usuario (administrador, usuario normal).
* **Protección de Datos**:
  + La aplicación implementará HTTPS para asegurar la transmisión de datos sensibles y proteger la información de los usuarios.

## 4.2. Escalabilidad

* **Despliegue en la Nube**:
  + Se desplegarán los componentes en Google Cloud Platform (GCP), aprovechando su escalabilidad y servicios gestionados como Cloud SQL para PostgreSQL.

## 4.3. Mantenimiento

* **Desarrollo Ágil**:
  + La metodología de desarrollo ágil se utilizará para gestionar el proyecto, facilitando la adaptación a cambios y la entrega continua de valor al cliente.
  + Se llevarán a cabo reuniones diarias y revisiones periódicas del progreso.
* **Documentación**:
  + La documentación del código y de las API se generará automáticamente y se mantendrá actualizada para facilitar el mantenimiento y la incorporación de nuevos desarrolladores al equipo.

## 4.4. Integración Continua y Despliegue

* **Pipeline de CI/CD**:
  + Se implementará un pipeline de CI/CD que automatizará el proceso de pruebas, construcción y despliegue de la aplicación.

# 5. Estrategia de Implementación de IA

## 5.1. Recopilación de Datos

* **Datos de Usuario**:
  + Se recopilarán datos relevantes sobre las interacciones de los usuarios, incluyendo historial de partidos y sus preferencias.
  + Se establecerán políticas claras sobre la recopilación y uso de datos para cumplir con las normativas de privacidad.

## 5.2. Desarrollo de Modelos de IA

* **Entrenamiento y Evaluación**:
  + Los modelos de machine learning se entrenarán utilizando datos históricos, y se evaluarán mediante métricas como precisión y recall.
  + Se implementarán técnicas de validación cruzada para garantizar la robustez de los modelos.

## 5.3. Integración en el Sistema

* **API de IA**:
  + Se creará una API que permita al backend consultar los modelos de IA en tiempo real, facilitando la obtención de recomendaciones personalizadas para los usuarios.

## 

# 6. Conclusión

La arquitectura propuesta para **"FlashMatch"** proporciona una base sólida para el desarrollo de una aplicación innovadora que no solo conecta a jugadores de fútbol, sino que también ofrece una experiencia personalizada a través de recomendaciones impulsadas por inteligencia artificial. La implementación de tecnologías modernas, prácticas de desarrollo ágil y un enfoque centrado en la seguridad y la escalabilidad garantizará que **"FlashMatch"** sea un producto competitivo en el mercado.