



# Desarrollo de Quickmatch: Backend & Frontend

Este informe presenta el progreso realizado en el desarrollo del backend y frontend de la plataforma Quickmatch, incluyendo la elección de tecnologías, resolución de problemas, y los siguientes pasos a seguir.

# Quickmatch: Tecnologías Escogidas

## Lenguajes

Java, JavaScript, y CSS.

## Framework

SpringBoot.

## Data

SQL / NoSQL.

## ORM

JPA (Java Persistence API).

```
C:\Users\miche\OneDrive\Escritorio\DATOS2\QuickMart-Backend>docker run -p 8080:8080 quickmart

:: Spring Boot ::
(v3.3.2)

2024-08-16T16:22:05.131Z INFO 1 --- [QuickMart] [main] com.ufm.QuickMart.QuickMartApplication : Starting QuickMartApplication v0.0.1-SNAPSHOT using Java 17.0.2 with PID 1 (/app/QuickMart.jar started by root in /app)
2024-08-16T16:22:05.139Z INFO 1 --- [QuickMart] [main] com.ufm.QuickMart.QuickMartApplication : No active profile set, falling back to 1 default profile: "default"
2024-08-16T16:22:06.606Z INFO 1 --- [QuickMart] [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port 8080 (http)
2024-08-16T16:22:06.625Z INFO 1 --- [QuickMart] [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2024-08-16T16:22:06.625Z INFO 1 --- [QuickMart] [main] o.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/10.1.26]
2024-08-16T16:22:06.686Z INFO 1 --- [QuickMart] [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2024-08-16T16:22:06.688Z INFO 1 --- [QuickMart] [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1382 ms
2024-08-16T16:22:07.105Z INFO 1 --- [QuickMart] [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8080 (http) with context path '/'
2024-08-16T16:22:07.134Z INFO 1 --- [QuickMart] [main] com.ufm.QuickMart.QuickMartApplication : Started QuickMartApplication in 2.697 seconds (process running for 3.597)
2024-08-16T16:29:43.942Z INFO 1 --- [QuickMart] [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherServlet'
2024-08-16T16:29:43.942Z INFO 1 --- [QuickMart] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2024-08-16T16:29:43.944Z INFO 1 --- [QuickMart] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 2 ms

C:\Users\miche\OneDrive\Escritorio\DATOS2\QuickMart-Backend>
```

# Primera Etapa: Selección y Resolución

1

## 1. Selección de Lenguajes

Se seleccionaron los lenguajes Java, JavaScript, y CSS para la construcción de la plataforma.

2

## 2. Framework & ORM

Se eligió SpringBoot como framework principal y JPA para la gestión de la base de datos.

3

## 3. Problemática con PATHS

Se encontró un problema con la gestión de rutas en el backend, resolviendo el acceso a archivos.

# Estructura del Backend

## Config

Configuración de la aplicación, incluyendo la configuración de la base de datos, el servidor, y otros aspectos.

## Controllers

Controladores que reciben las peticiones HTTP del frontend y las procesan.

## DTO

Objetos de transferencia de datos que se utilizan para intercambiar información entre el backend y el frontend.

## Entities

Representación de los datos en la base de datos.

## Repositories

Interfaz para interactuar con la base de datos.

## Services

Lógica de negocio que realiza las operaciones con los datos.

# Pruebas & Desarrollo en Docker



## Docker

Docker se utiliza para el desarrollo, pruebas, y despliegue de la plataforma.

```
≡ .env
⊘ .gitignore
🐳 docker-compose.yml
🐳 Dockerfile
📄 mvnw
≡ mvnw.cmd
m pom.xml
≡ pom.xml.versionsBackup
M↓ README.md
```

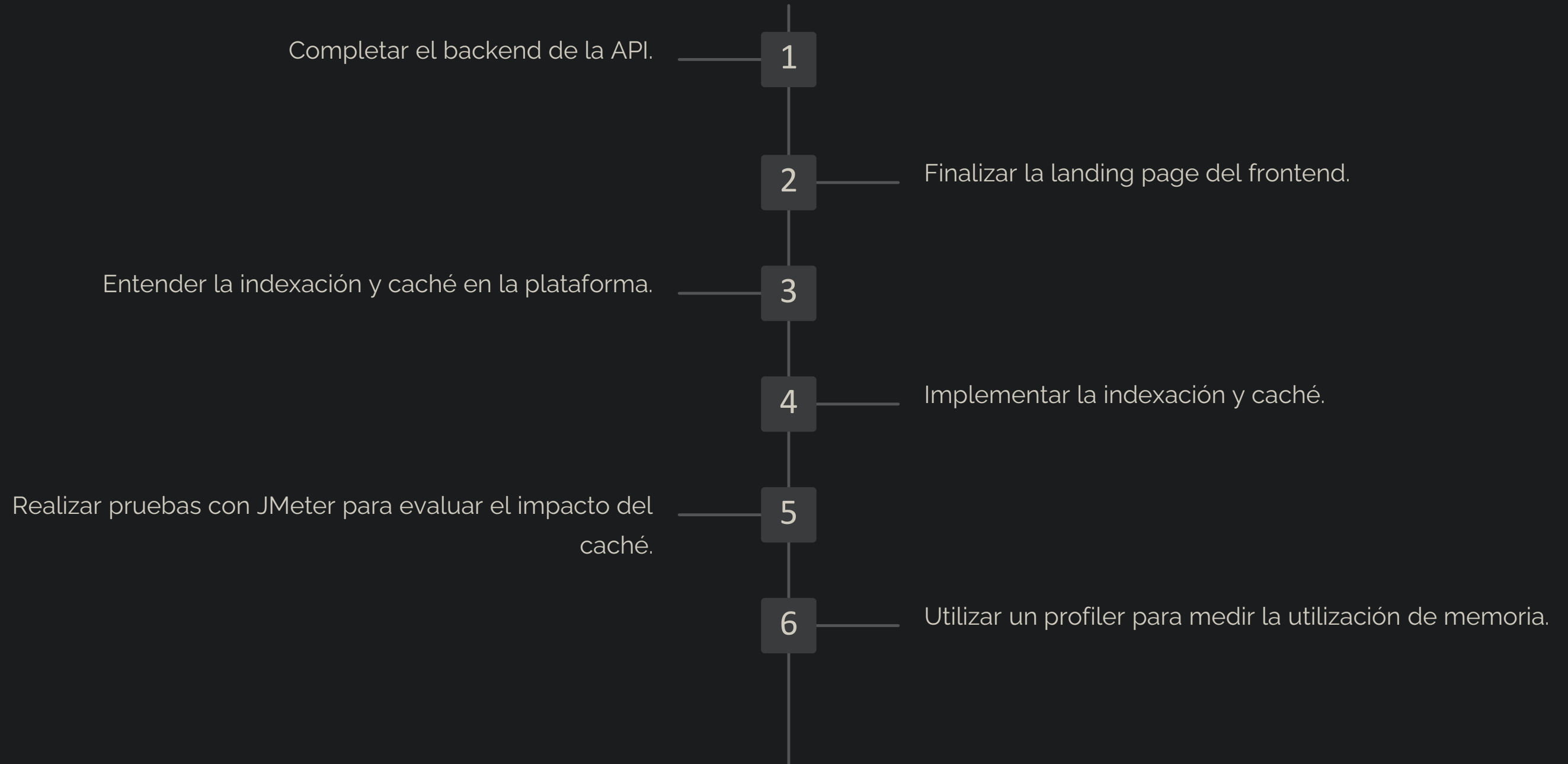


## Postman

Postman se utiliza para probar las API del backend.

```
▼ QuickMatch
  POST predicciones
  POST agregar usuario 4 a grupo 4
  POST crear grupo con id de torneo, {i...
  PUT actualizar puntos
  PUT Actualizar resultado de partido
  GET torneos
  GET EquiposXtorneo
  GET predicciones(usuarioid,grupoid...
  GET predicciones(usuarioid, partid...
  GET predicciones(usuarioid,grupoid...
  GET predicciones(partidoid,grupoid...
  GET grupos de un usuario
  GET tabla con orden descendente
  GET Ver partidos por id de torneo y...
  DEL Eliminar grupo por id
```

# Next Steps: Hitos Importantes



# Uso de @Caché: Optimización de Rendimiento

76

Sin @Caché

Tiempo de carga inicial.

27

Con @Caché

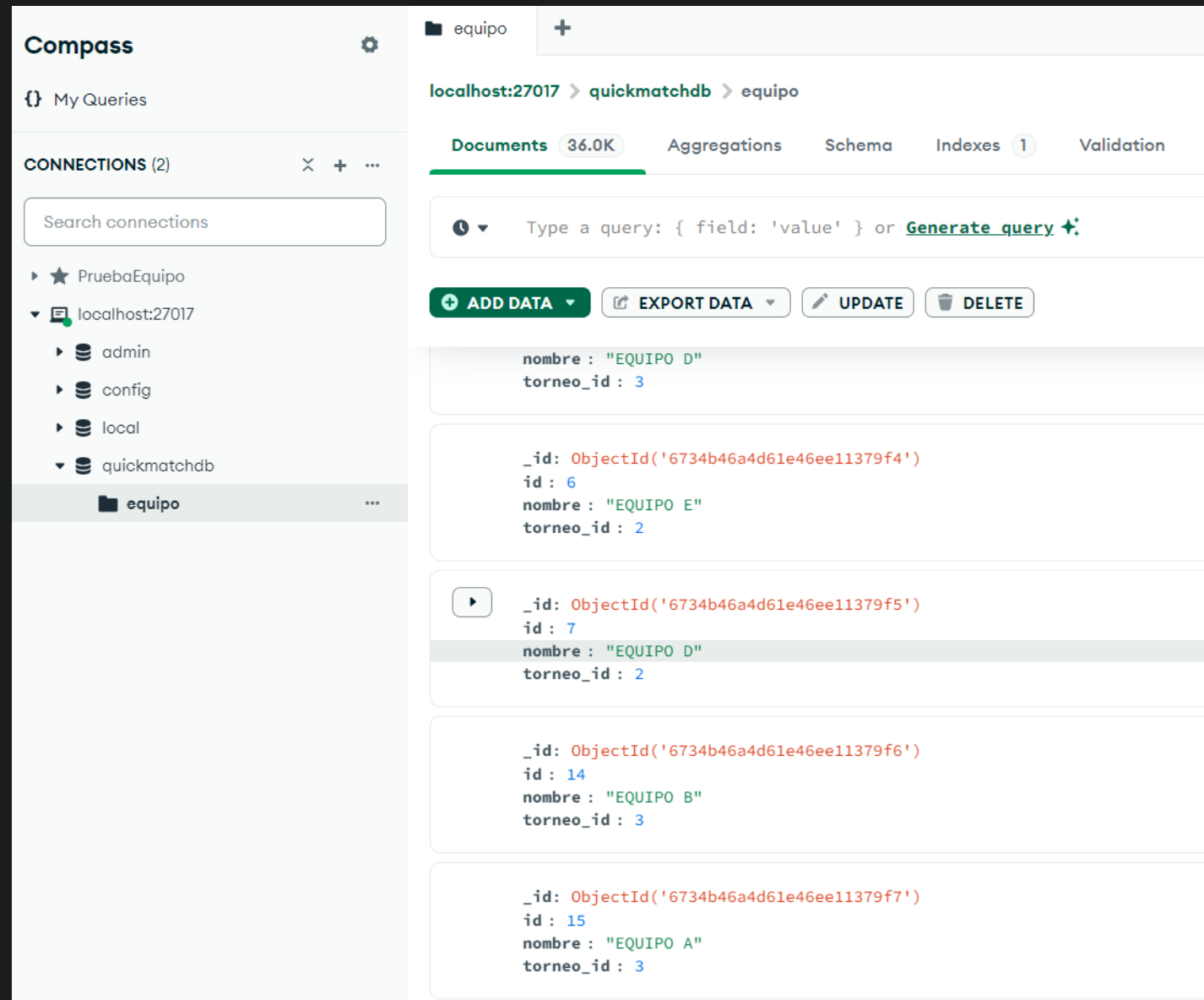
Tiempo de carga con caché.

```
Nombre del hilo:EquiposxTorneo 1-1
Comienzo de muestra:2024-11-10 19:26:27 CST
Tiempo de carga:27
Connect Time:1
Latencia:27
Tamaño en bytes:1119
Sent bytes:196
Headers size in bytes:421
Body size in bytes:698
Conteo de muestra:1
Conteo de error:0
Data type ("text"|"bin"|""):text
Código de respuesta:200
Mensaje de respuesta:
```

```
HTTPSampleResult campos:
ContentType: application/json
DataEncoding: null
```



# Nosql: Evolución hacia MongoDB



Nombre del hilo:Grupo de Hilos 1-1  
Comienzo de muestra:2024-11-13 08:25:13 CST  
Tiempo de carga:135  
Connect Time:51  
Latencia:127  
Tamaño en bytes:829  
Sent bytes:200  
Headers size in bytes:421  
Body size in bytes:408  
Conteo de muestra:1  
Conteo de error:0  
Data type ("text"|"bin"|""):text  
Código de respuesta:200  
Mensaje de respuesta:

HTTPSampleResult campos:  
ContentType: application/json  
DataEncoding: null

La primera llamada es de 135 (vs 192 usando MySQL)