Portal para comunidades de propietarios

José Luis Zorita Gutiérrez

Grado de Ingeniería Informática

ÍNDICE

Contexto

Contexto bajo el que se desarrolla el proyecto

Existen aplicaciones para que los propietarios puedan consultar información, pero suelen contar con varias deficiencias, como por ejemplo:

- Las aplicaciones actuales muestran **información muy limitada.**
- El administrador **no utiliza estas plataformas, trabaja desde su ERP.**
- **Falta de interacción entre el administrador y el propietario.**
- Los propietarios siguen haciendo sus **gestiones de forma presencial, por vía telefónica o carta.**

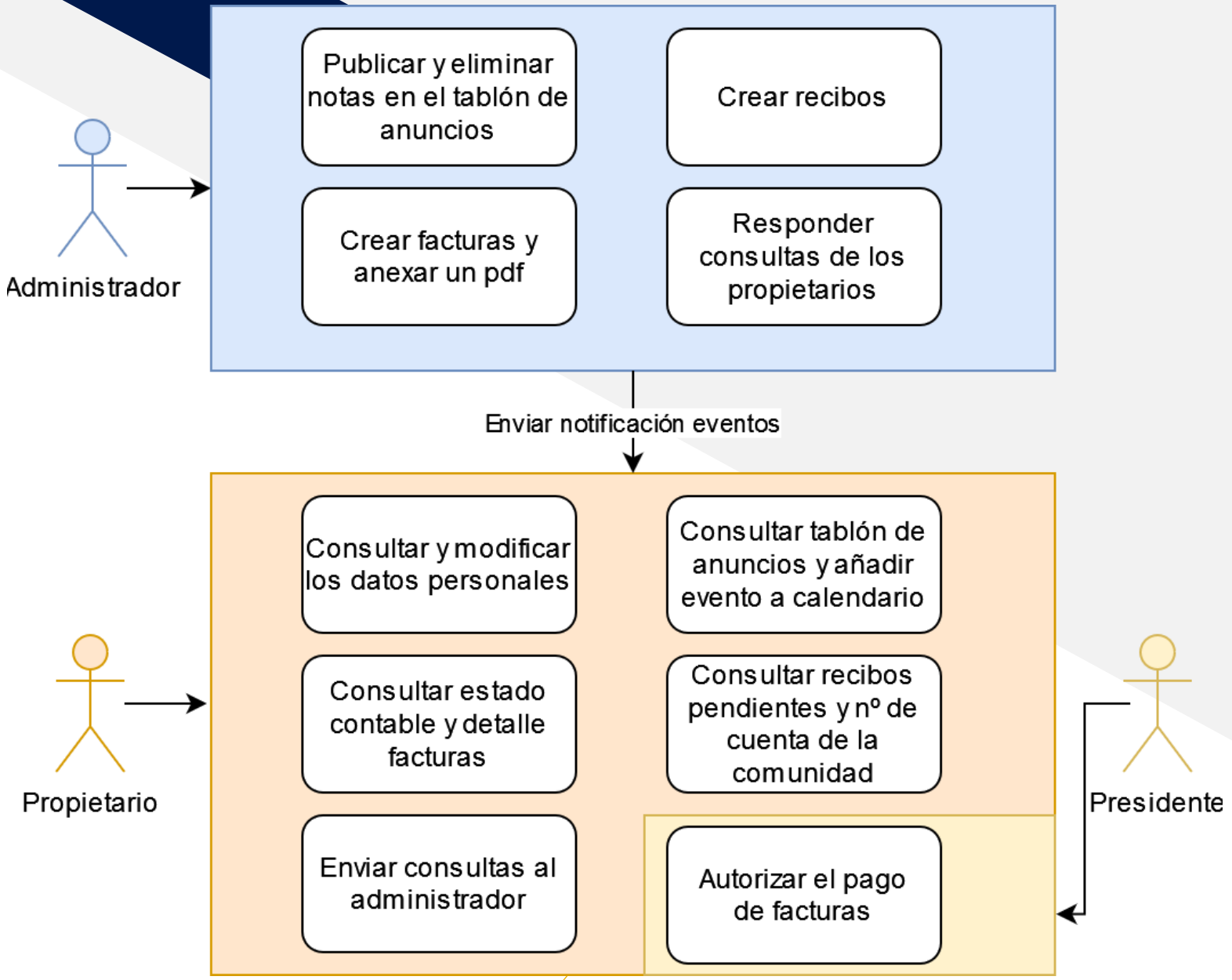
Objetivos del trabajo

El proyecto tiene los siguientes objetivos

Crear una plataforma mediante microservicios que ofrezca servicios API REST a consumir desde un cliente web que permita:

- Que los **vecinos** y el **administrador** estén en el **mismo ecosistema**.
- Crear un **canal de comunicación** entre los propietarios y el administrador.
- Al propietario realizar acciones como **consultar el estado contable**, **autorizar facturas** o **cambiar sus datos**.
- Al administrador realizar acciones como **crear facturas y recibos**.

Usuarios y requisitos



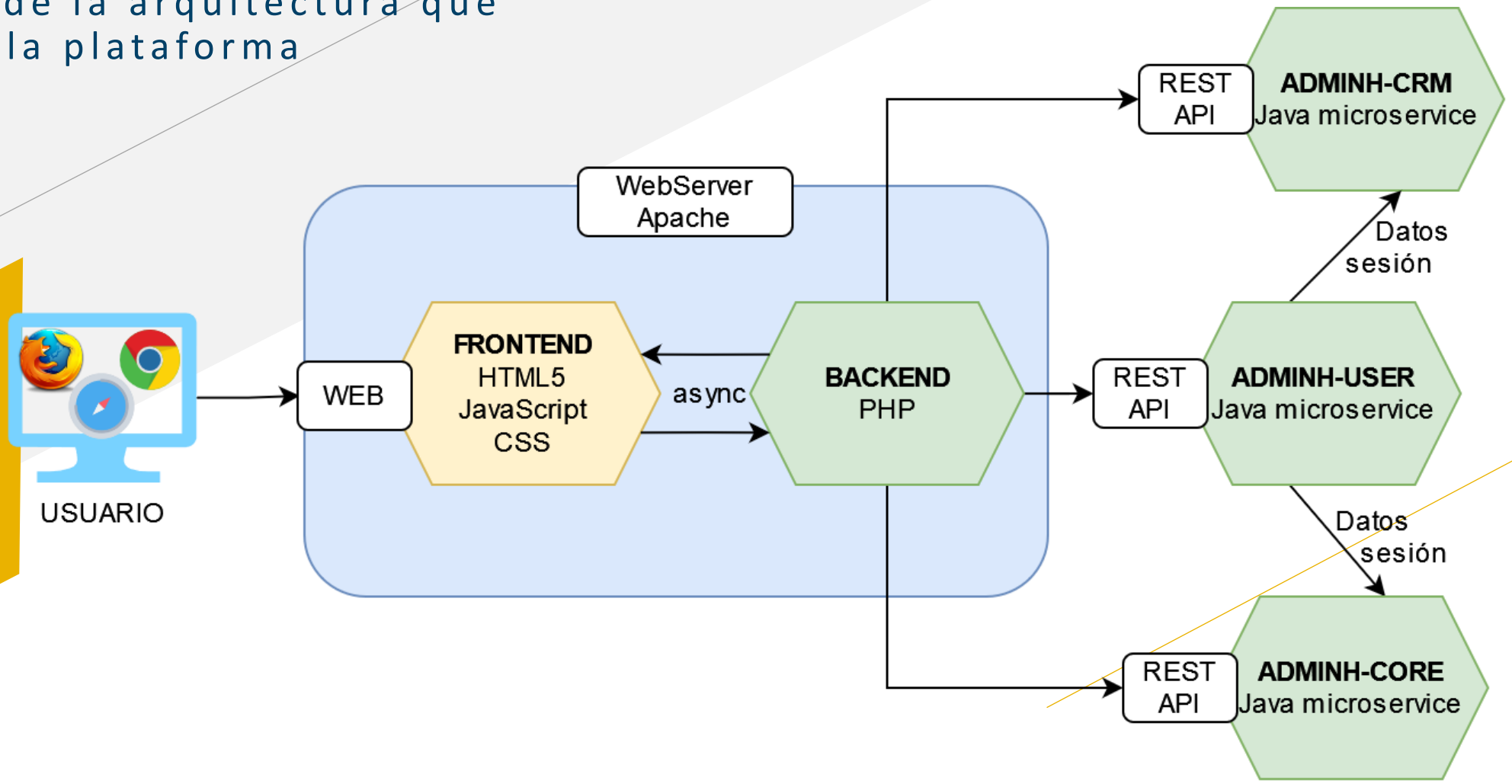
Prototipo

Se ha creado un prototipo con **FIGMA** que permite inspeccionar el estilo css para trasladarlo a la aplicación final

[Enlace prototipo]

Arquitectura

Diagrama de la arquitectura que albergará la plataforma



Detalle tecnologías utilizadas

Microservicios

Propósito	Aplicación
Lenguaje / kit de desarrollo	OpenJDK
Gestión del proyecto	Maven
Persistencia / manipulación datos	spring
Base de datos	PostgreSQL
Comunicación entre microservicios	Apache kafka
Automatización proyecto	spring boot
Librería anotaciones	Project Lombok
Inicialización base de datos	HIBERNATE

Aplicación web

Propósito	Aplicación
Paquete que integra servidor apache y php	XAMPP
Lenguajes para codificación de *frontend*	HTML CSS
Lenguaje de programación lado cliente	Javascript JS
Programación lado servidor	php
Comunicación microservicios	curl

Entornos de desarrollo

Propósito	Aplicación
Codificación aplicación web y ficheros Docker	Visual Studio Code
Codificación microservicios	IntelliJ IJ

Entidades Microservicios

Microservicios

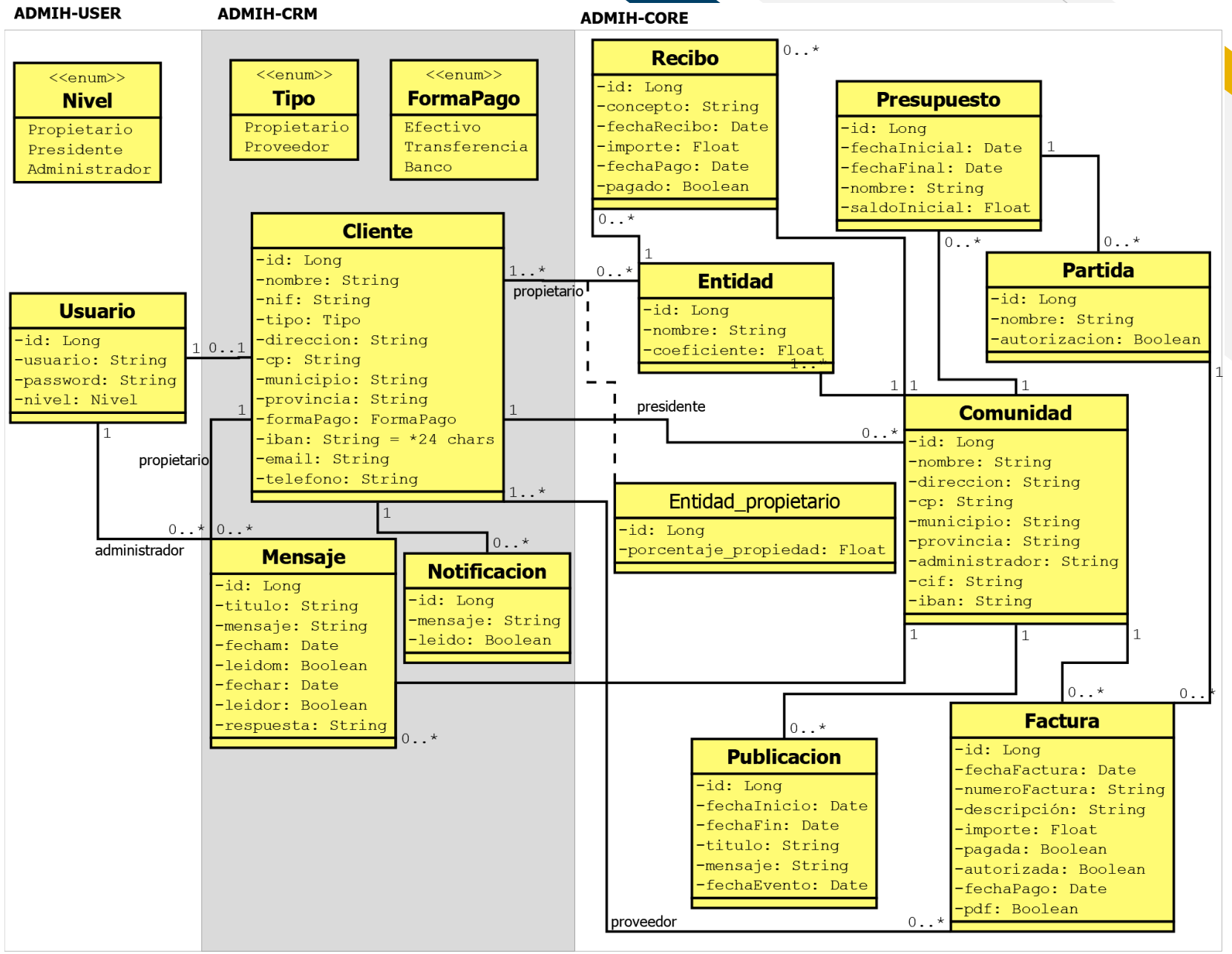
Entidades de los microservicios que conforman las tablas de sus bases de datos

Adminh-user: **db-user**

Adminh-crm: **db-crm**

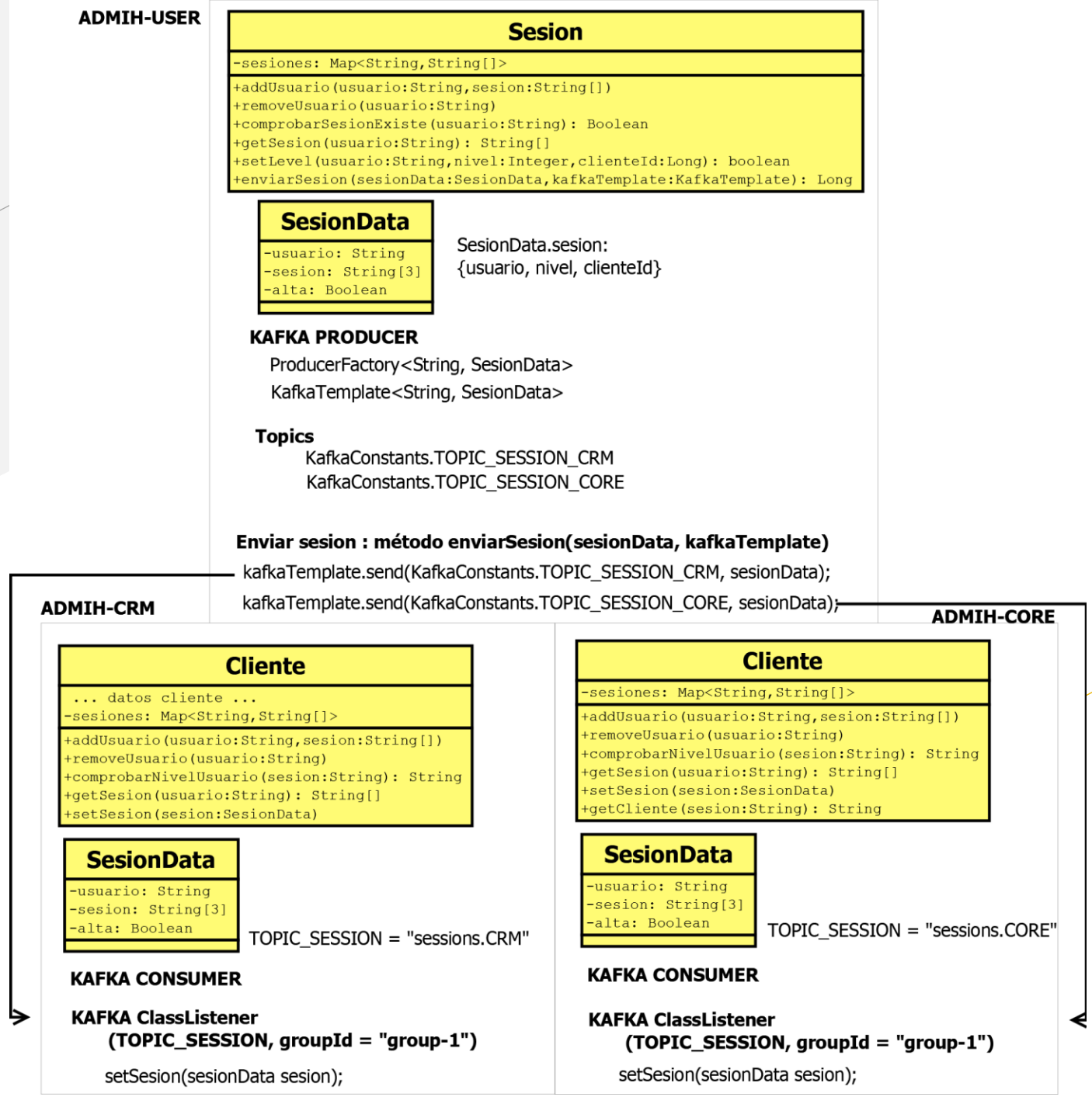
adminh-core: **db-core**

Implementado en etapa 2



Comunicación Kafka

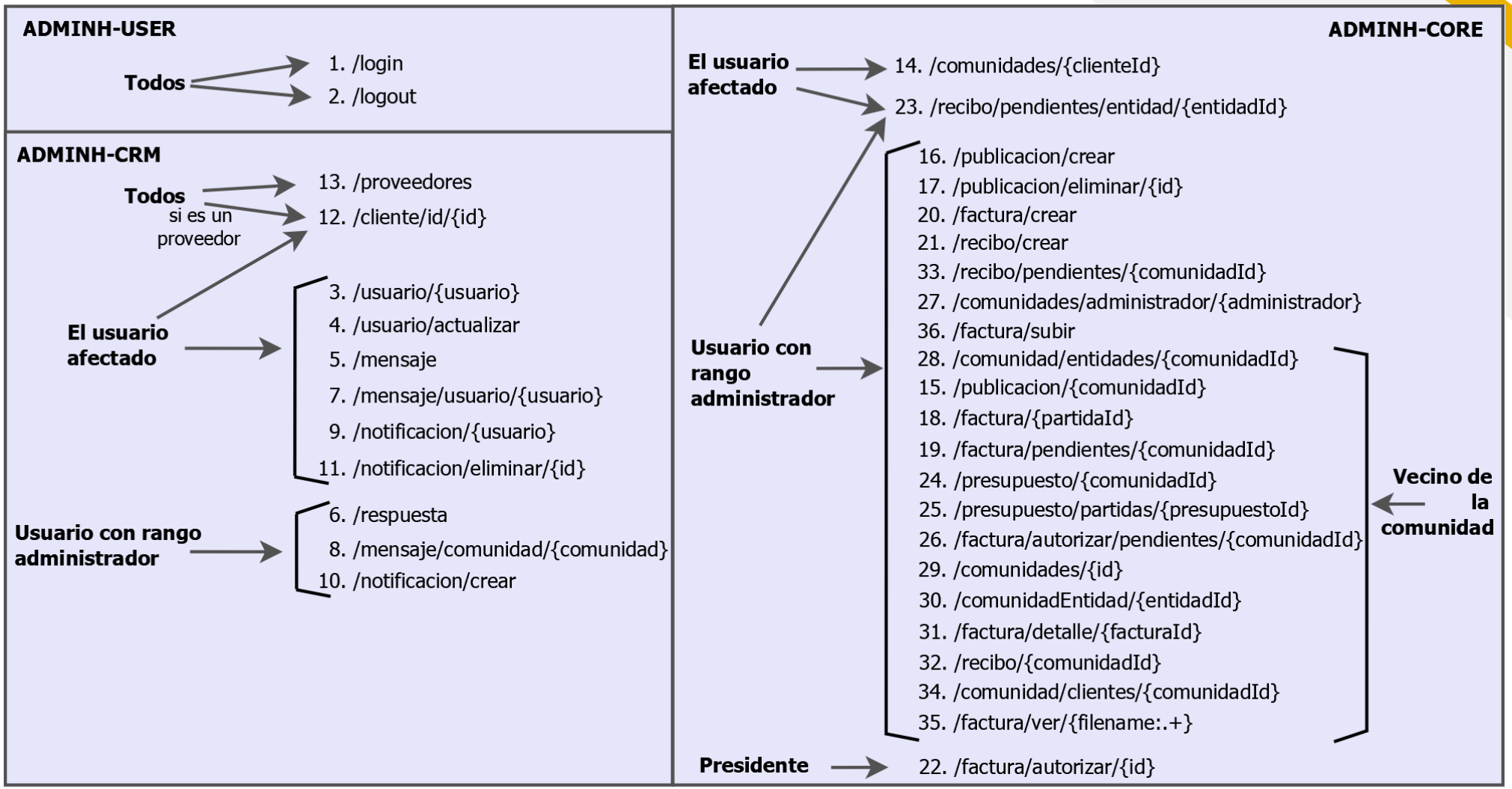
Diagrama comunicación entre microservicios

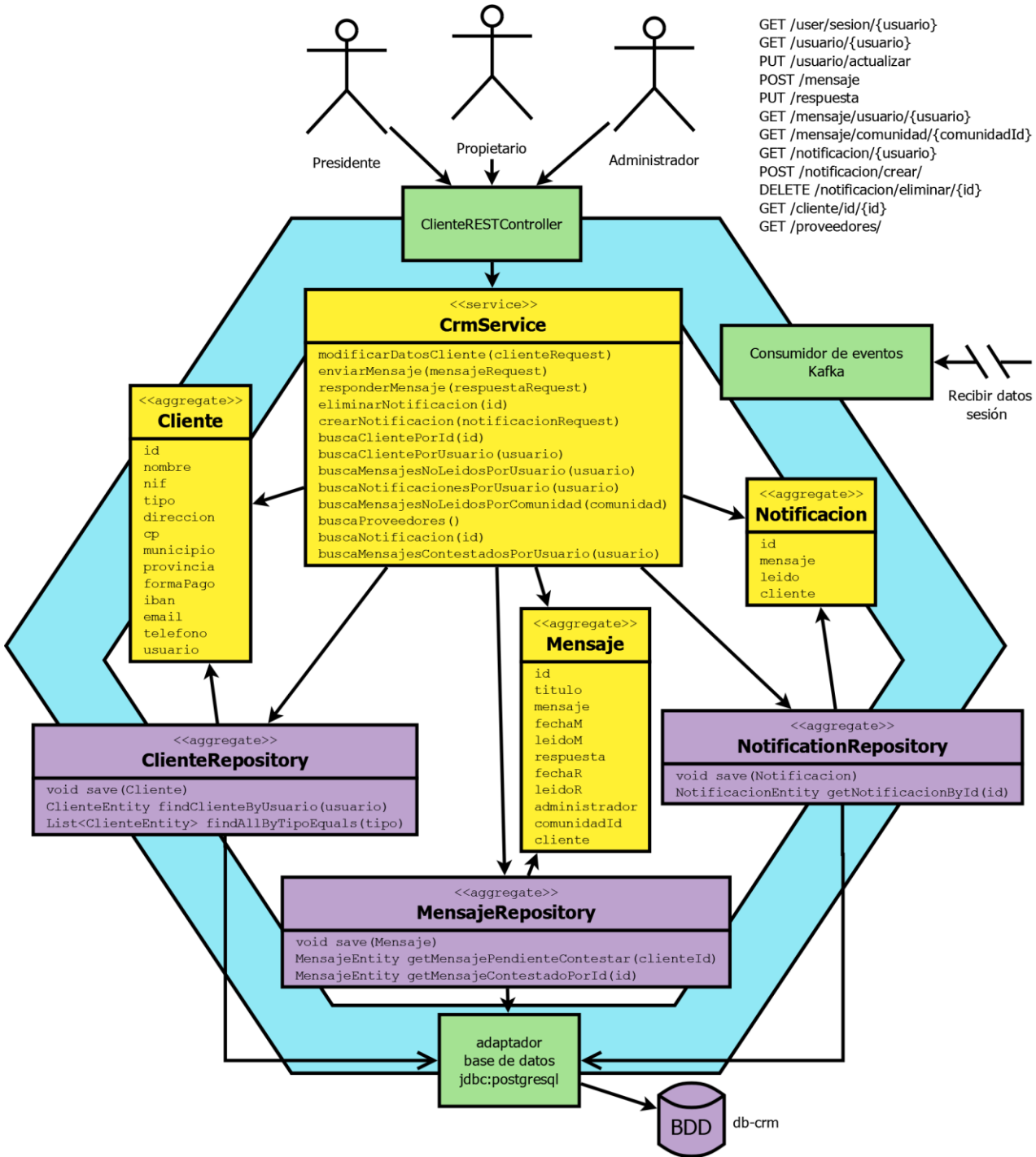


API REST

Servicios API REST Microservicios

Métodos de los servicios API REST y permisos de usuario para su ejecución



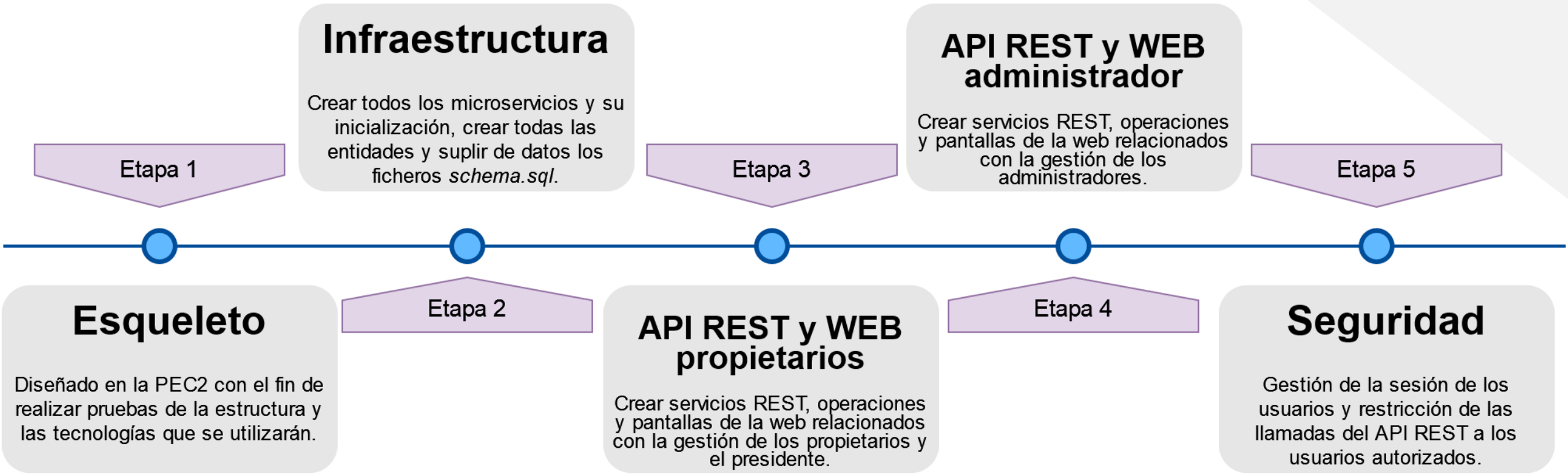


Arquitectura hexagonal

Arquitectura hexagonal adminh-crm

Ejemplo de la arquitectura seguida por el microservicio adminh-crm.

Etapas implementación



Esqueleto

Etapa 1 de la implementación

```
[+] Running 7/7
 - Container tfg-db-user-1       Healthy
 - Container tfg-zookeeper-1     Started
 - Container tfg-adminer-1       Started
 - Container tfg-db-crm-1        Started
 - Container tfg-kafka-1         Healthy
 - Container tfg-adminh-user-1   Started
 - Container tfg-adminh-crm-1    Started
PS C:\TFG>
```

Arranque microservicios con docker-compose

Creado un esqueleto de forma previa a la PEC 3 con el objetivo de:

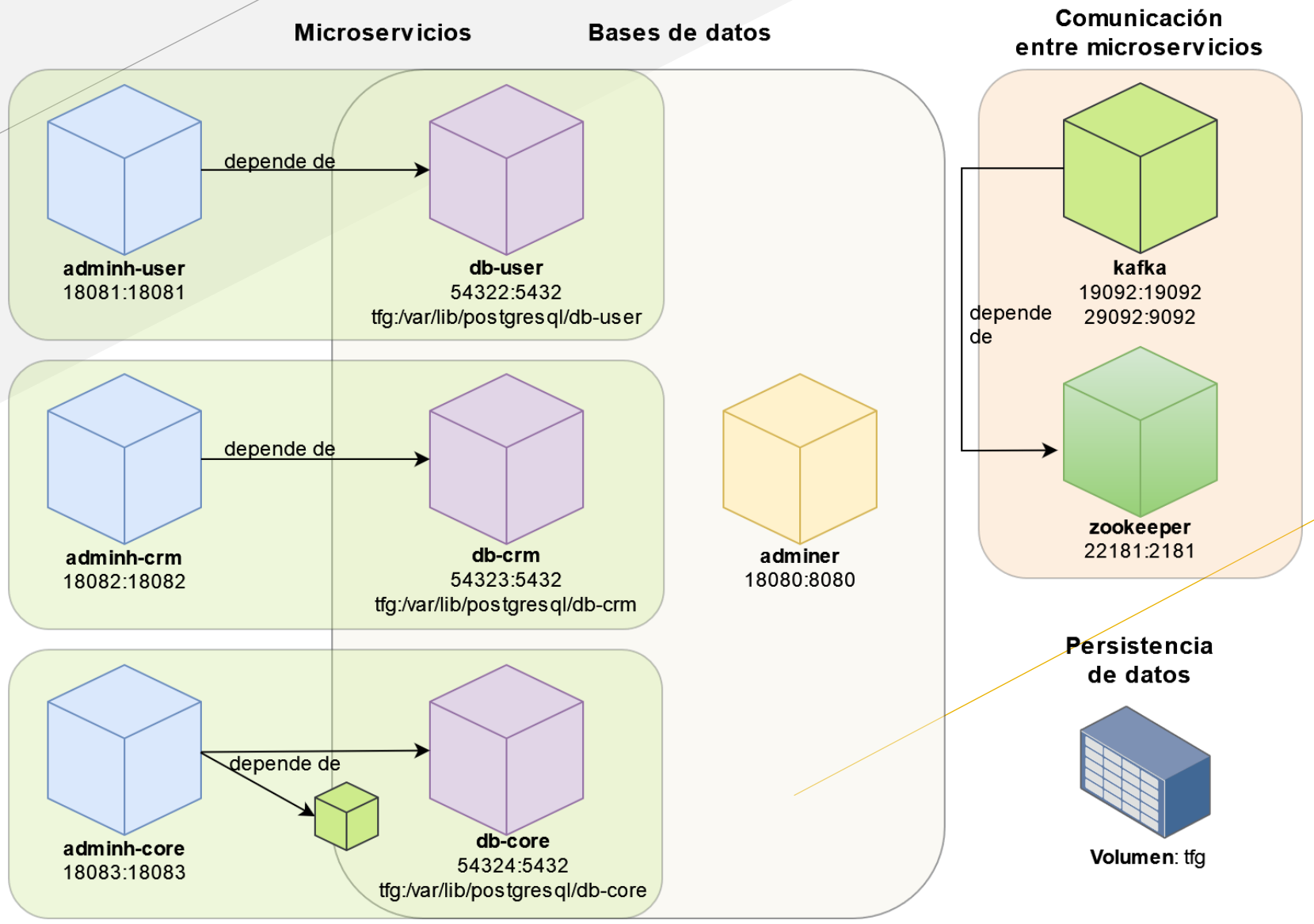
- Realizar una **configuración inicial del entorno.**
- **Comparar posibles tecnologías** a utilizar para su aprobación final.
- **Probar las herramientas** que se utilizarán en el posterior desarrollo.
- **Dejar el sistema preparado** para iniciar la fase de desarrollo.

Infraestructura

Etapa 2 de la implementación

Objetivos de esta etapa:

- Acabar de **configurar toda la infraestructura**.
- **Crear las entidades** en cada microservicio según el esquema de entidades.
- **Suplir a los ficheros *schema.sql* de información** para inicializar con datos la aplicación.

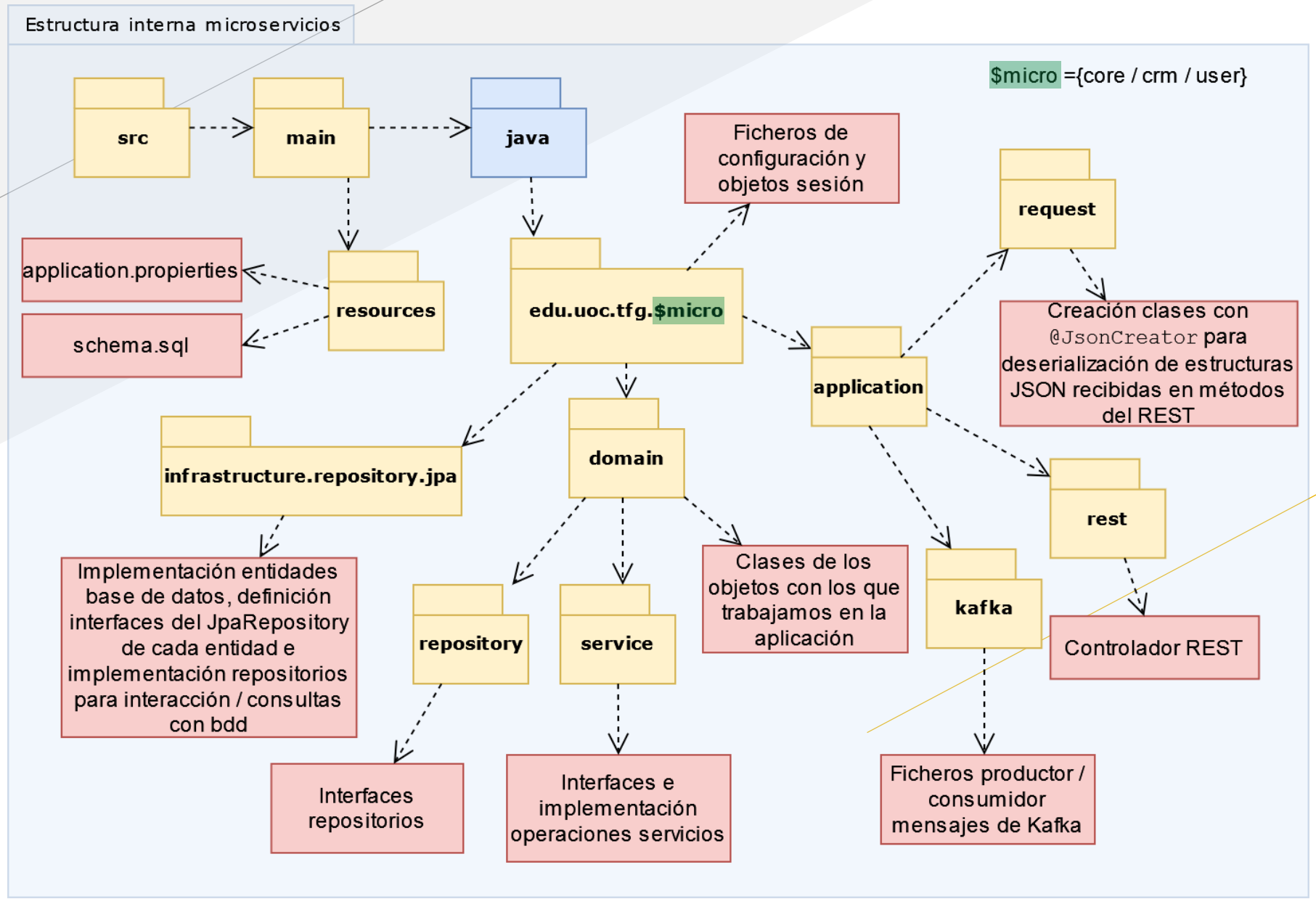


Servicios definidos en `docker-compose.yml`

API REST y WEB - Microservicios

Etapas 3 y 4 de la implementación

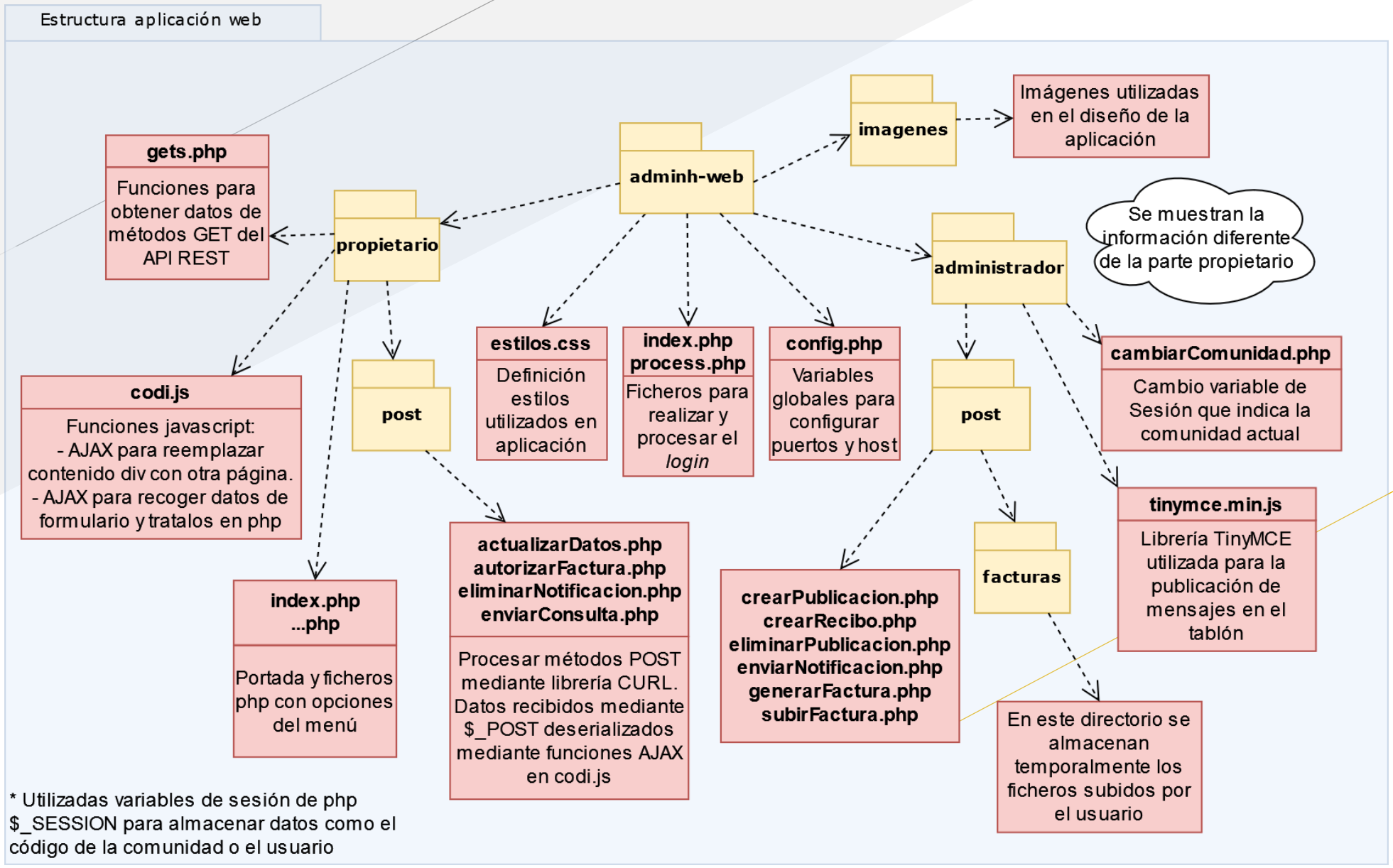
Se muestra en rojo el detalle de la codificación realizada



API REST y WEB - Aplicación web

Etapas 3 y 4 de la implementación

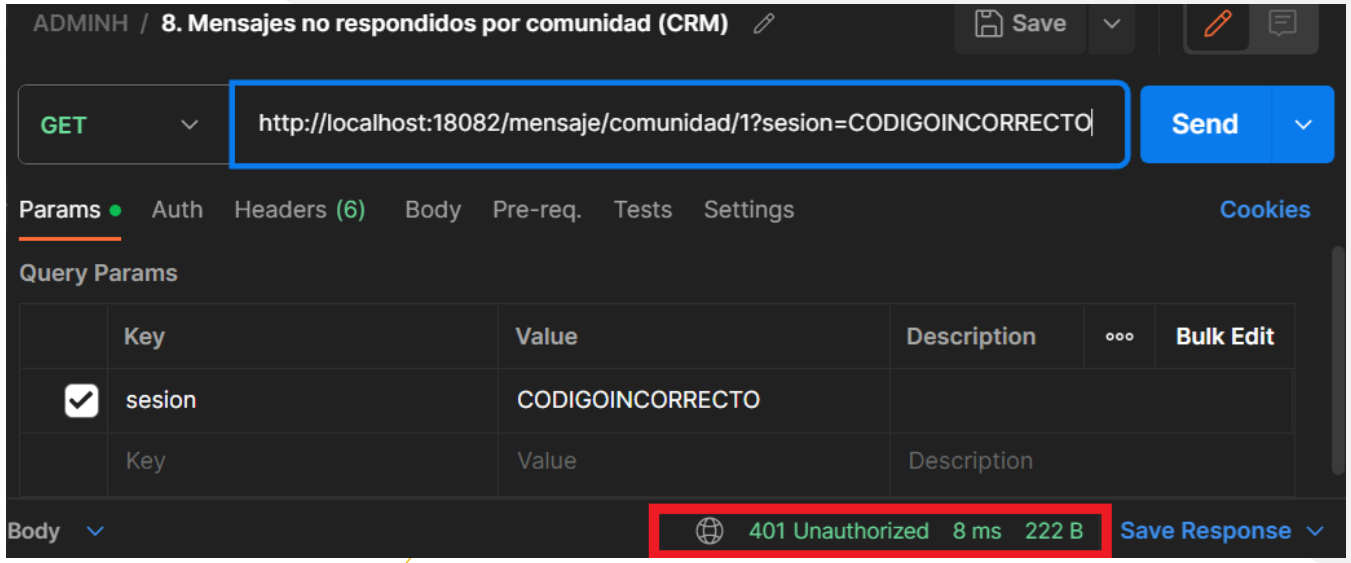
Se muestra en rojo el detalle de la codificación realizada



Seguridad

Etapa 5 de la implementación

- Modificación de todos los **métodos del API REST** para **verificar el código de sesión** y **que sólo puedan ser ejecutados por usuarios autorizados**.
- Ha supuesto la necesidad de **crear nuevas operaciones para realizar verificacaciones.**
- **Código 401 UNAUTHORIZED en caso de no contar con permisos.**



Problemas y dificultades

Dificultades encontradas durante la implementación de la plataforma

- **Error de compilación** al producirse un bucle infinito por relación entrelazada entre entidades al obtener datos [`toDomain()`].
- Error a la hora de **insertar datos** si en las instrucciones INSERT INTO del `schema.sql` se especificaba manualmente un identificador.
- En aplicación web error por **falta de permisos** de escritura en directorio en el que el se suben temporalmente los ficheros.

Conclusiones

- ☑ **Valoración global positiva**: se han alcanzado objetivos y la aplicación se asemeja a prototipo de Figma.
- ☑ **Infraestructura configurada antes de la fase de la implementación** debido a la configuración de un esqueleto de forma previa.
- ☑ **Profundización** en diferentes tecnologías para el desarrollo de microservicios.
- ☑ **Aprendizaje de nuevas herramientas** y lenguajes para desarrollo web.

- ☒ **Gran desviación respecto a la planificación inicial.**
- ☒ **Falta de pruebas en el código.**
- ☒ **No se han utilizado *frameworks* para la aplicación web**.