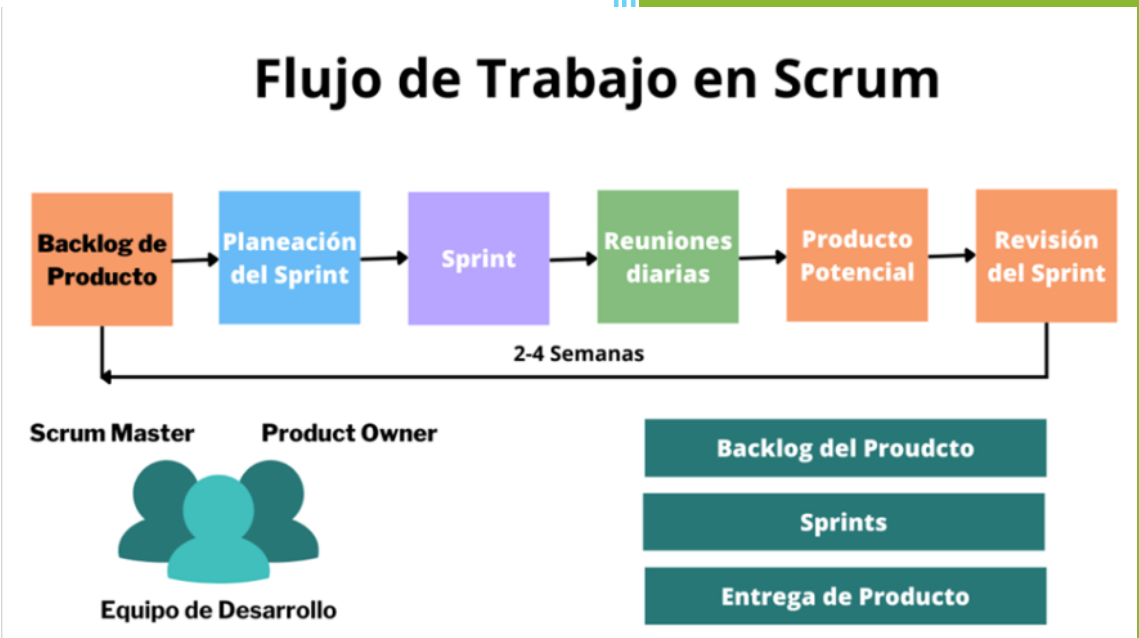
PR1.2 HACKATHON SPRINT2



Tamara Simón López
María José Molina Toboso

2ºDAMV PROYECTO

MOVING ARROUND THE CITY

App Move Albacete – muévete por la ciudad

PROTOTIPADO CON HERRAMIENTAS DIGITALES

1. DIAGRAMA DE CASOS DE USO (UML)

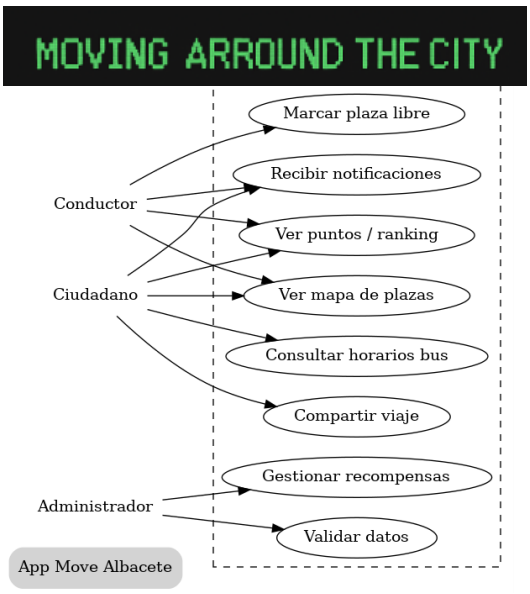
Este diagrama representa como interactúan los usuarios (actores) con la app.

Los actores principales y sus interacciones serán:

- **Conductor** → buscar aparcamiento, marcar plaza libre, ver puntos, recibir notificaciones, ver mapa de plazas.
- **Ciudadano/Usuario** → consultar horarios bus, compartir viaje para movilidad sostenible, recibir notificaciones, ver puntos, ver mapa de plazas.
- **Administrador** → validar datos, gestionar recompensas, revisar incidencias.

Casos de uso clave:

- Ver el mapa con las plazas libres.
- Marcar plazas libres.
- Consultar horarios de autobús.
- Crear/Unirse a un viaje compartido.
- Ver puntos y ranking.
- Recibir notificaciones.



2. MODELADO DE DATOS

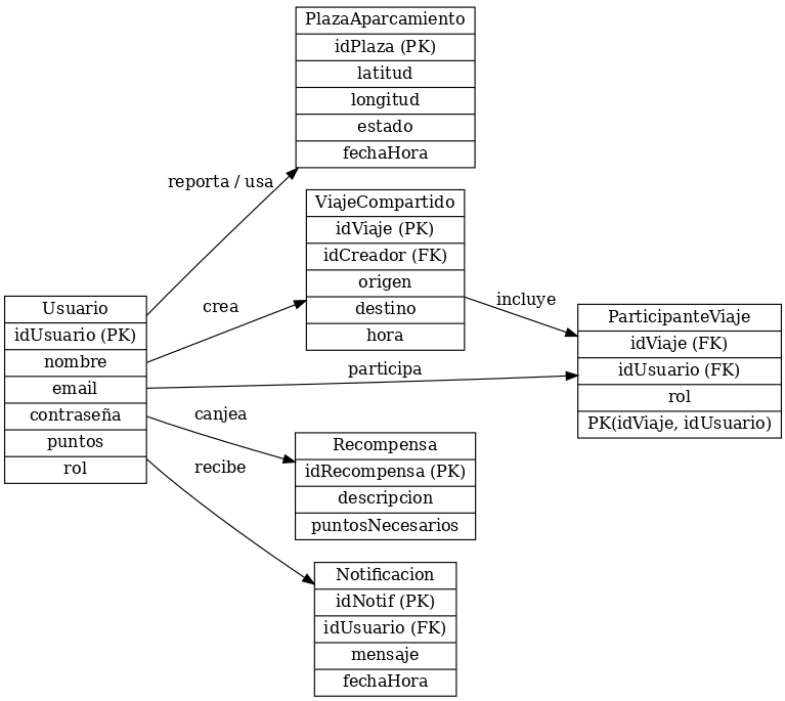
Hay que sacar un **modelo entidad/relación (E/R)** y luego convertirlo a **relacional**.

Las entidades principales son:

- **Usuario** (id, nombre, email, puntos, rol).
- **PlazaAparcamiento** (id, ubicaciónGPS, estado [libre/ocupada], últimaActualización).
- **Notificación** (id, mensaje, idUsuario, fechaHora).
- **ViajeCompartido** (id, origen, destino, hora, idCreador).
- **ParticipanteViaje** (idViaje, idUsuario, rol [conductor/pasajero]).
- **Recompensa** (id, descripción, puntosNecesarios).

Modelo relacional:

- **Usuario**(idUsuario PK, nombre, email, contraseña, puntos, rol)
- **PlazaAparcamiento**(idPlaza PK, latitud, longitud, estado, fechaHora)
- **ViajeCompartido**(idViaje PK, idCreador FK→Usuario, origen, destino, hora)
- **ParticipanteViaje**(idViaje FK→ViajeCompartido, idUsuario FK→Usuario, rol, PRIMARY KEY(idViaje, idUsuario))
- **Recompensa**(idRecompensa PK, descripcion, puntosNecesarios)
- **Notificación**(idNotif PK, idUsuario FK→Usuario, mensaje, fechaHora)

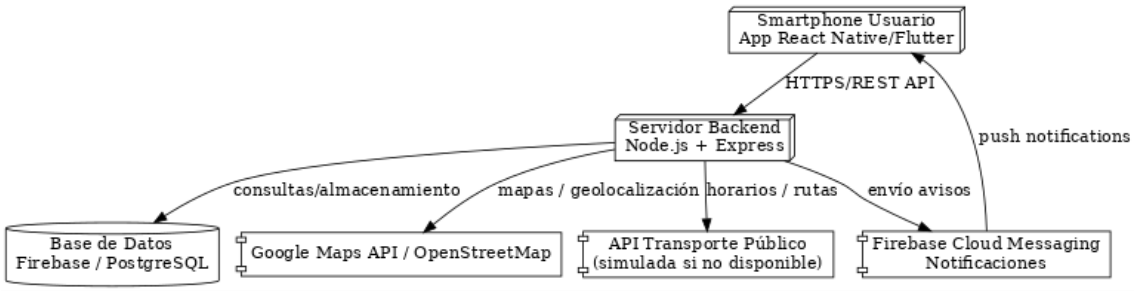


3. DIAGRAMA DE DESPLIEGUE (UML)

Muestra la arquitectura de hardware/software donde corre la app.

Move Albacete:

- **Móvil Usuario (App React Native / Flutter)**(internet/https)
- **Servidor Backend (Node.js + Express)**
    - API REST (plazas, bus, viajes, puntos).
    - conectado a **Base de Datos (Firebase / PostgreSQL)**.
- **Servicios externos**:
    - Google Maps API / OpenStreetMap.
    - API Transporte Público (o datos simulados).
- **OneSignal / Firebase Cloud Messaging** → para notificaciones.



4. TECNOLOGÍAS QUE SE VAN A USAR

Frontend (App móvil):

- **React Native** → multiplataforma (Android/iOS) con un solo código.
- Alternativa: **Flutter**

Backend:

- **Node.js + Express** → ligero, rápido y fácil de integrar con Firebase o SQL.
- API REST para móviles.

Base de Datos:

- **Firebase Realtime DB / Firestore** → ideal para actualizaciones en tiempo real de plazas.
- **PostgreSQL** → SQL estructurado con relaciones (viajes, recompensas, etc.).

Mapas y geolocalización:

- **Google Maps API** o **OpenStreetMap (gratuito y open source)**.

Gamificación y notificaciones:

- **Firebase Cloud Messaging (FCM)** para notificaciones.
- Sistema de puntos gestionado en la base de datos.

¿POR QUÉ HEMOS ESCOGIDO ESTAS TECNOLOGÍAS?

- Multiplataforma (llegar a Android e iOS con un solo código).
- Tiempo real (Firebase / Websockets para plazas).
- Integración sencilla con mapas y transporte público.
- Escalabilidad (PostgreSQL para datos relacionales más complejos).

¿PARA QUÉ?

- **React Native/Flutter:** entregar una app usable en móviles.
- **Node.js:** centralizar la lógica y las APIs.
- **Firebase/PostgreSQL:** guardar usuarios, plazas, viajes, puntos.
- **Mapas API:** mostrar ubicaciones en tiempo real.
- **Notificaciones:** mejorar la experiencia y avisar de plazas libres o buses cercanos.