

Integrantes: Carlos Caramé. Cristian Morales. Iván Roldán

# TAREA 2 – Aplicaciones Híbridas (AH)

## Bloque 1 – Proyecto base Ionic + Capacitor

- Crear proyecto Ionic Angular y verificar con *ionic serve*.
- Configurar rutas iniciales (Home) y estructura con ion-router-outlet.
- Sincronizar con Capacitor, abrir en Android Studio y probar en dispositivo/emulador.

## Bloque 2 – UI principal

- Diseñar cabecera con ion-header + ion-toolbar.
- Maquetar listado con ion-list / ion-item / ion-card.
- Añadir búsqueda con ion-searchbar.
- Implementar navegación lista → detalle (/detalle/:id).

## Bloque 3 – Datos y lógica

- Crear DataService con datos mock + carga de detalle.
- Crear formulario con ngForm + [(ngModel)], validación y lógica add/update.

## Bloque 4 – Persistencia local

- Instalar SQLite/Preferences.
- Diseñar e implementar repositorio CRUD local.
- Sincronizar DataService con repositorio y prueba offline.

## Bloque 5 – Firebase + API externa

- Crear proyecto Firebase e instalar firebase + angular/fire.
- Crear FirebaseService CRUD y sincronización remota.

- Configurar HttpClientModule + ApiService y consumir API externa.
- Añadir botón Sincronizar + manejo de errores.

## **Bloque 6 – Funciones de dispositivo**

- Instalar Geolocation y Camera.
- Crear DeviceService.
- Guardar lat/long y foto en SQLite/Firebase.
- Mostrar ubicación (mapa o enlace).
- Implementar swipe para eliminar + feedback háptico.

## **Bloque 7 – PWA**

- Añadir soporte PWA.
- Configurar manifest (nombre, iconos).
- Pruebas de instalación + modo offline (assets cache).

---

# **TAREA 3 – Programación Multimedia y Dispositivos Móviles (PMDM)**

## **Bloque 1 – Navegación y pantallas**

- Configurar NavController + NavHost.
- Crear pantallas principales (lista, detalle, formulario).
- Implementar navegación: lista → detalle → atrás.

## **Bloque 2 – Listado + UI**

- Crear lista mock con ItemCard y ListScreen (LazyColumn).

- Añadir estilos: scroll, paddings, colores.
- Mostrar imagen y datos en detalle.

### **Bloque 3 – Formulario Compose**

- UI con TextField + botones.
- Estados con rememberSaveable.
- Validación + mensajes de error.
- Guardar datos con ViewModel y navegación al listar.

### **Bloque 4 – Persistencia Room**

- Crear entidad, DAO CRUD y RoomDatabase.
- Añadir repositorio + Flow con ViewModel.
- Conectar UI → Room y guardar datos del formulario.

### **Bloque 5 – Multimedia**

- Mostrar imágenes locales (galería/cámara).
- Reproducir audio (SoundPool, MediaPlayer, ExoPlayer).
- Reproducir vídeo local/remoto con ExoPlayer + PlayerView Compose.
- Añadir controles y liberar recursos.

### **Bloque 6 – Animaciones**

- Animaciones implícitas (scale/alpha).
- Animación de visibilidad.
- Animación infinita.
- updateTransition y Animatable para swipes.

### **Bloque 7 – Sensores + Maps**

- Obtener SensorManager y registrar listener.
  - Mostrar datos en tiempo real, detectar shake y cambiar UI por luminosidad.
  - Integrar Maps Compose, mostrar coordenadas, marcadores y centrar cámara.
  - Botón para alternar pantalla/mapa.
- 

## **TAREA 4 – PSP (Servicios y Procesos)**

### **Bloque 1 – Backend base**

- Crear proyecto Spring Boot + entidad + repositorio JPA.
- Crear servicio de lógica + validaciones + DTOs/Mappers.
- Controlador REST completo con manejo global de errores.

### **Bloque 2 – Base de datos**

- Configurar H2 para desarrollo.
- Probar repositorio con datos y API con Postman.
- Añadir perfil Postgres (Render).

### **Bloque 3 – Despliegue**

- Crear repo Git y subir a GitHub.
- Crear BBDD en Render y desplegar servicio.
- Añadir variables de entorno y probar API pública.

### **Bloque 4 – Integración con la app**

- Configurar cliente HTTP.
- Implementar GET/POST/PUT/DELETE.

- Manejo de errores + estados UI (Loading/Error/Success).