Taller 2 - Introducción al Desarrollo Web Móvil

Grupo 1 — Equipo: SmartCoders

Integrantes

- Bastian Salinas — 21.848.994-K
- Benjamín Cuello — 21.682.135-1
- Benjamín Salas — 21.758.667-4
- Tomás Guerra — 21.664.344-5

Descripción General

InfoMóvil ahora integra un ecosistema con **3 servicios backend propios** y un **frontend móvil empaquetado como aplicación Android (APK)** usando **Apache Cordova**.

La app mantiene las mismas funcionalidades del Taller 1 (Pokémon, países, clima y feriados), pero ahora **todos los datos provienen de nuestras propias APIs y bases de datos**.

Objetivos del Proyecto

- Desarrollar **3 APIs independientes**, cada una en una tecnología distinta.
- Crear un **frontend móvil responsivo** que consuma dichas APIs.
- **Empaquetar la app como APK Android** funcional.
- Mantener el mismo diseño, navegación y estructura modular del Taller 1.

APIs a usar:

API 1 — Pokémon (NestJS + PostgreSQL)

- **Endpoints**
  - `GET /pokemon?limit&offset` → lista de pokemones
  - `GET /pokemon/:nombreOId` → detalle de un pokemon
- **Datos almacenados:** id, nombre, sprites, tipos, altura, peso, estadísticas.
- **Puerto:** `http://localhost:3000`

API 2 — Países (Express + PostgreSQL)

- **Endpoints**
  - `GET /countries` → lista de todos los países
  - `GET /countries/search?name=Chile` → búsqueda por nombre
- **Datos:** nombre común, oficial, bandera, región, capital, población, código ISO.

- **Puerto:** `http://localhost:4000`

API 3 — Clima y Feriados (FastAPI + MongoDB)

- **Endpoints**
  - `GET /weather?city=La%20Serena` → devuelve temperatura y viento actuales
  - `GET /holidays/{countryCode}/{year}` → devuelve feriados del país y año indicado
- **Datos:**
  - Clima → nombre, latitud, longitud, temperatura, viento.
  - Feriados → fecha, nombre local y nombre oficial.
- **Puerto:** `http://localhost:8000`

Frontend (Cordova + HTML + JS + Tailwind)

- Código fuente ubicado en `frontend/www/`, estructurado para reutilizarse dentro de Cordova (`www/`).
- Mantiene el diseño y la lógica del Taller 1, ahora parametrizado para consumir nuestras APIs o datos mock.
- **Configuración:** `frontend/www/js/config.js` define el modo y las URLs base:

```
var MODE = 'public' // 'public' | 'local' | 'mock'
var BASE_URL_POKEMON = 'http://localhost:3000'
var BASE_URL_COUNTRIES = 'http://localhost:4000'
var BASE_URL_FASTAPI = 'http://localhost:8000'
```

- El encabezado muestra y permite cambiar el modo de datos (APIs públicas/locales/mocks), recordando la última selección.
- Consulta `frontend/README.md` para instrucciones específicas (Docker, servidores estáticos, Cordova).
- Consulta `docs/backend-contracts.md` para implementar las APIs con el contrato correcto.
- El modo se puede cambiar también vía `?mode=mock` o `?mode=local` en la URL, útil para demos.
- Requiere Node.js >= 16 para ejecutar los scripts opcionales (`npm run sync-www`).
- Los datos de ejemplo para `MODE='mock'` están en `frontend/www/mock/`.

Tecnologías a utilizar:

Frontend

- HTML5
- CSS3 + Tailwind CSS (CDN)
- JavaScript ES6
- Apache Cordova

Backend

- NestJS (TypeScript) + PostgreSQL
- Express (Node.js) + PostgreSQL
- FastAPI (Python) + MongoDB

Estructura del Repositorio de momento

```
infomovil-taller2/
├── backend/
│   ├── express/
│   │   └── Dockerfile
│   ├── fastapi/
│   │   └── Dockerfile
│   ├── nest/
│   │   └── Dockerfile
│   └── README.md
├── frontend/
│   ├── Dockerfile
│   ├── README.md
│   └── www/
│       ├── index.html
│       ├── styles.css
│       ├── js/
│       │   ├── api.js
│       │   ├── config.js
│       │   ├── config.sample.js
│       │   ├── countries.js
│       │   ├── holidays.js
│       │   ├── main.js
│       │   ├── pokemon.js
│       │   ├── ui.js
│       │   ├── utils.js
│       │   └── weather.js
│       └── mock/
│           ├── countries_all.json
│           ├── holidays_CL_2025.json
│           ├── pokemon/
│           │   ├── bulbasaur.json
│           │   ├── charizard.json
│           │   ├── charmander.json
│           │   ├── gengar.json
│           │   ├── ivysaur.json
│           │   ├── mew.json
│           │   ├── squirtle.json
│           │   ├── pikachu.json
│           │   └── snorlax.json
│           │   └── venusaur.json
│           ├── pokemon_list.json
│           ├── weather_Calama.json
│           ├── weather_La_Serena.json
│           ├── weather_Santiago.json
```

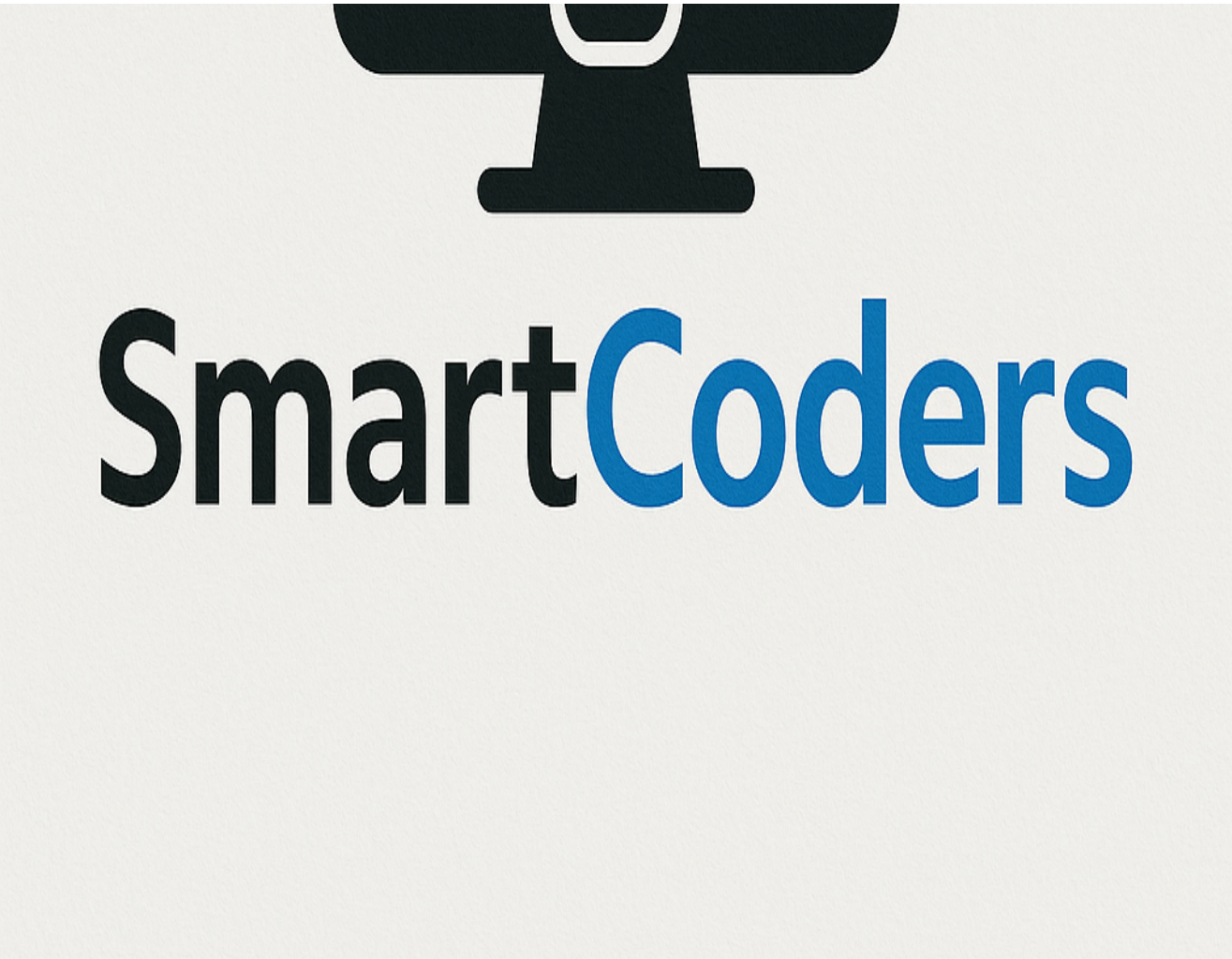
```
│           └── weather_default.json
├── scripts/
│   └── sync-www.js
├── docs/
│   ├── backend-contracts.md
│   └── frontend-testing-checklist.md
├── docker-compose.yml
└── README.md
```

Backend (pendiente)

- Cada servicio tiene su carpeta dentro de `backend/` con un `Dockerfile` placeholder y documentación en `backend/README.md`.
- Al implementar las APIs se debe actualizar ese `Dockerfile` y añadir la configuración de la base de datos correspondiente.
- Los contenedores exponen puertos `3000`, `4000` y `8000`, que también se publican en el host.

Organización

División de tareas y organización:
https://docs.google.com/spreadsheets/d/1SS0sQna__lw2i_N7hFHYkcZuJKCjdoBD3aPRTeOkSSM/edit?gid=0#gid=0



Proyecto desarrollado para el curso Introducción al Desarrollo Web Móvil - Universidad Católica del Norte (UCN) 2025

Despues para Cordova:

- En emulador Android: usar `http://10.0.2.2:<puerto>` como host para `BASE_URL_*`.
- En Docker: `http://host.docker.internal:<puerto>`