

Backend Contracts

Estos contratos resumen lo que el frontend espera recibir de cada API propia. Los ejemplos se basan en los mocks ubicados en frontend/www/mock/.

Pokémon (NestJS)

- **GET** `/pokemon?limit=&offset=` → listado con estructura similar a la PokeAPI:

```
{
  "count": 10,
  "results": [{ "name": "bulbasaur" }, { "name": "ivysaur" }]
}
```

- **GET** `/pokemon/:idOrName` → objeto con los campos utilizados por el frontend:

```
{
  "id": 25,
  "name": "pikachu",
  "height": 4,
  "weight": 60,
  "types": [{ "slot": 1, "type": { "name": "electric" } }],
  "sprites": {
    "front_default": "https://.../25.png",
    "back_default": "https://.../back/25.png",
    "other": {
      "official-artwork": {
        "front_default": "https://.../official-artwork/25.png"
      }
    }
  },
  "stats": [
    { "base_stat": 35, "stat": { "name": "hp" } },
    { "base_stat": 55, "stat": { "name": "attack" } }
  ]
}
```

Países (Express)

- **GET** `/countries` → array de países ordenables en memoria:

```
[
  {
    "name": { "common": "Chile", "official": "República de Chile" },
  }
]
```

```

    "flags": { "png": "https://flagcdn.com/w320/cl.png" },
    "region": "Americas",
    "capital": ["Santiago"],
    "population": 19116201,
    "cca2": "CL"
  }
]

```

- **GET** `/countries/search?name=Chile` → mismo arreglo filtrado por `name.common` (case-insensitive).

Clima y feriados (FastAPI)

- **GET** `/weather?city=Coquimbo` →

```

{
  "coordenadas": {
    "name": "Coquimbo",
    "country_code": "CL",
    "latitude": -29.95,
    "longitude": -71.34
  },
  "clima": {
    "current": {
      "temperature_2m": 16.4,
      "wind_speed_10m": 5.1
    }
  }
}

```

- **GET** `/holidays/{countryCode}/{year}` →

```

[
  { "date": "2025-01-01", "localName": "Año Nuevo", "name": "New Year's Day" },
  { "date": "2025-05-01", "localName": "Día del Trabajador", "name": "Labour Day" }
]

```

Notas

- Mantener los mismos nombres de campos que las APIs públicas para minimizar cambios en el frontend.
- Las bases de datos pueden persistir estos datos con normalización propia, pero deben exponer el JSON anterior.

- Ante errores, responder con códigos HTTP adecuados (4xx/5xx) y un cuerpo JSON { "message": "..."} para que el frontend pueda informar al usuario.