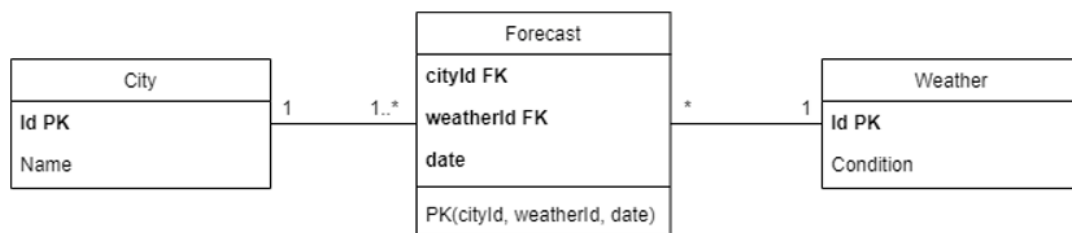Ferran Ramírez

Step 2 | API design (no code required)

To solve this part I've decided to create:

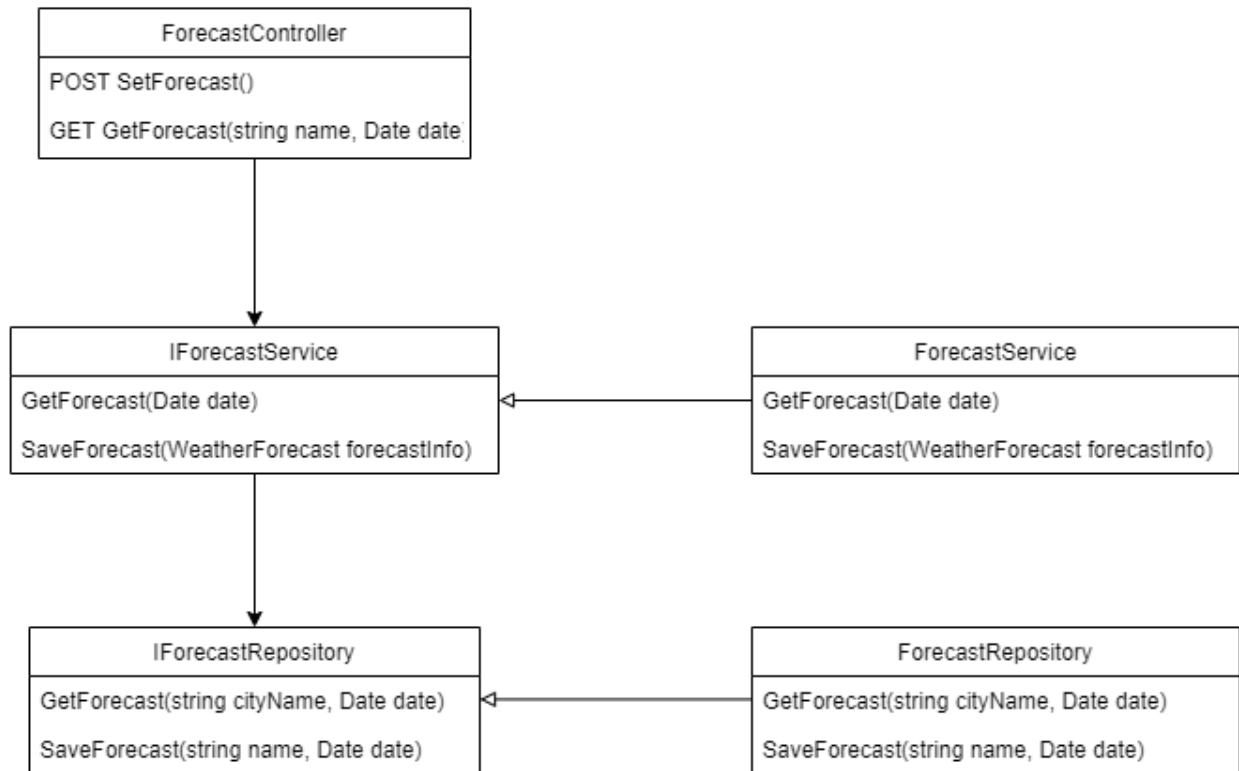## 1- Database

A database with the following tables:

City(id, name), **Weather**(id, condition) and **Forecast**(cityId, weatherId, date)

2- Project structure:

- A new **WebApi project**, where the *Forecast Controller* will be placed to capture the requests mentioned above.
- A new **Infrastructure Project**, where the repository classes will be placed, to interact with the database.

```
┌─────────────────────────────────────────┐
│              ForecastController          │
├─────────────────────────────────────────┤
│ POST SetForecast()                       │
│                                          │
│ GET GetForecast(string name, Date date)  │
└─────────────────────────────────────────┘
                    │
                    ▼
┌─────────────────────────────────────────┐        ┌─────────────────────────────────────────┐
│              IForecastService            │        │               ForecastService            │
├─────────────────────────────────────────┤        ├─────────────────────────────────────────┤
│ GetForecast(Date date)                   │◄───────│ GetForecast(Date date)                   │
│                                          │        │                                          │
│ SaveForecast(WeatherForecast forecastInfo)│       │ SaveForecast(WeatherForecast forecastInfo)│
└─────────────────────────────────────────┘        └─────────────────────────────────────────┘
                    │
                    ▼
┌─────────────────────────────────────────┐        ┌─────────────────────────────────────────┐
│             IForecastRepository          │        │             ForecastRepository           │
├─────────────────────────────────────────┤        ├─────────────────────────────────────────┤
│ GetForecast(string cityName, Date date)  │◄───────│ GetForecast(string cityName, Date date)  │
│                                          │        │                                          │
│ SaveForecast(string name, Date date)     │        │ SaveForecast(string name, Date date)     │
└─────────────────────────────────────────┘        └─────────────────────────────────────────┘
```

(For this design I'm ignoring the mapping details such as other things that might be considered during the development. I have also not included the the RequestServices that are already created in the code)

2- Endpoints

POST /forecast

It will require the city name and the date we want to save the weather from. It will send a request to the required external APIs to get the weather information for the given date and then this weather information will be stored in the database.

Payload:
```
{
        "City" : {city},
        "Day": {day}
}
```
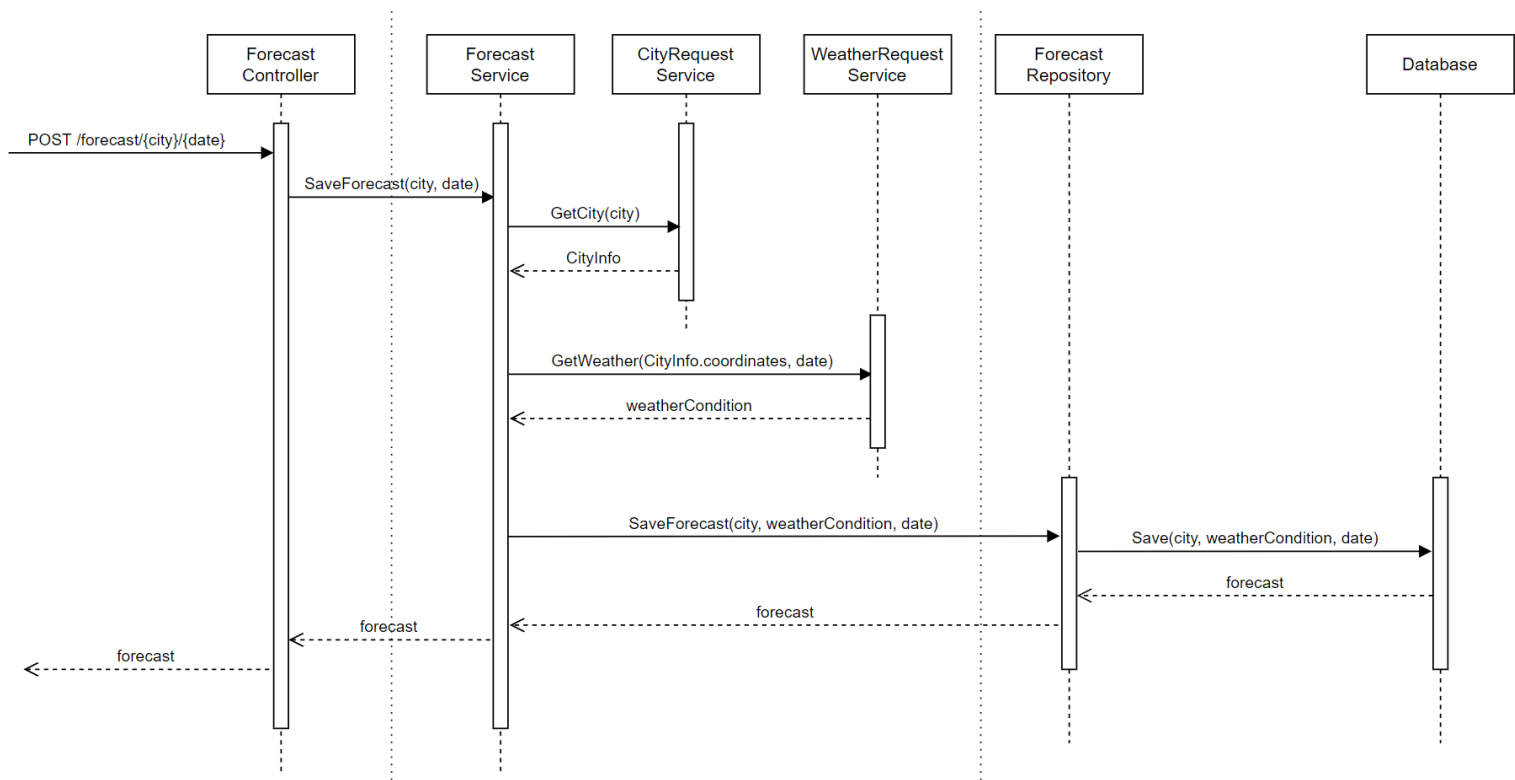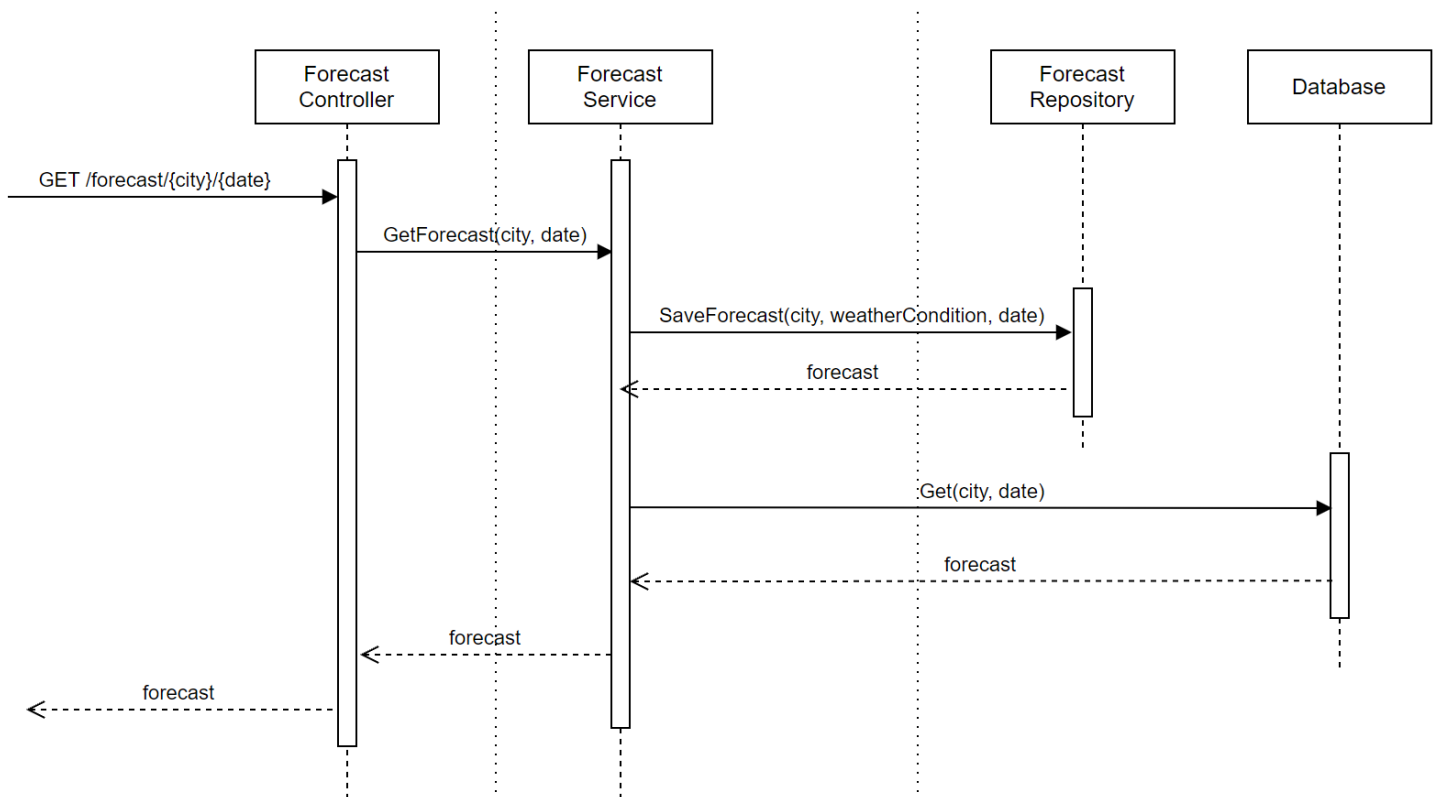
Responses:

200:

- The forecast has successfully been stored in the database.

  Example result:

  {
      "City" : "Amsterdam" ,
      "Day" : "20/01/2021",
      "WeatherCondition" : "Rainy"
  }

500:

- The given *{city}* could not be found.

- There was no weather forecast for the given *{day}*

GET /forecast/{city}/{day}

It will require the city name and the date we want the weather from. It will send a request to the database to get the weather information for the given date.

Responses:

200:

- The forecast has successfully been stored in the database.

  Example result:

  ```
  {
          "City" : "Amsterdam" ,
          "Day" : "20/01/2021",
          "WeatherCondition" : "Rainy"
  }
  ```

500:

- The given *{city}* could not be found in the database.
- There was no weather forecast in the database for the given *{day}*

GET /forecast/{city}/today

It will just require the city name. It will send a request to the database to get the weather information for the date the request is executed.
- The responses are the same as the **GET /forecast/{city}/{day}** call

GET /forecast/{city}/tomorrow

It will just require the city name. It will send a request to the database to get the weather information for the following day regarding the day the request is executed.
- The responses are the same as the **GET /forecast/{city}/{day}** call.