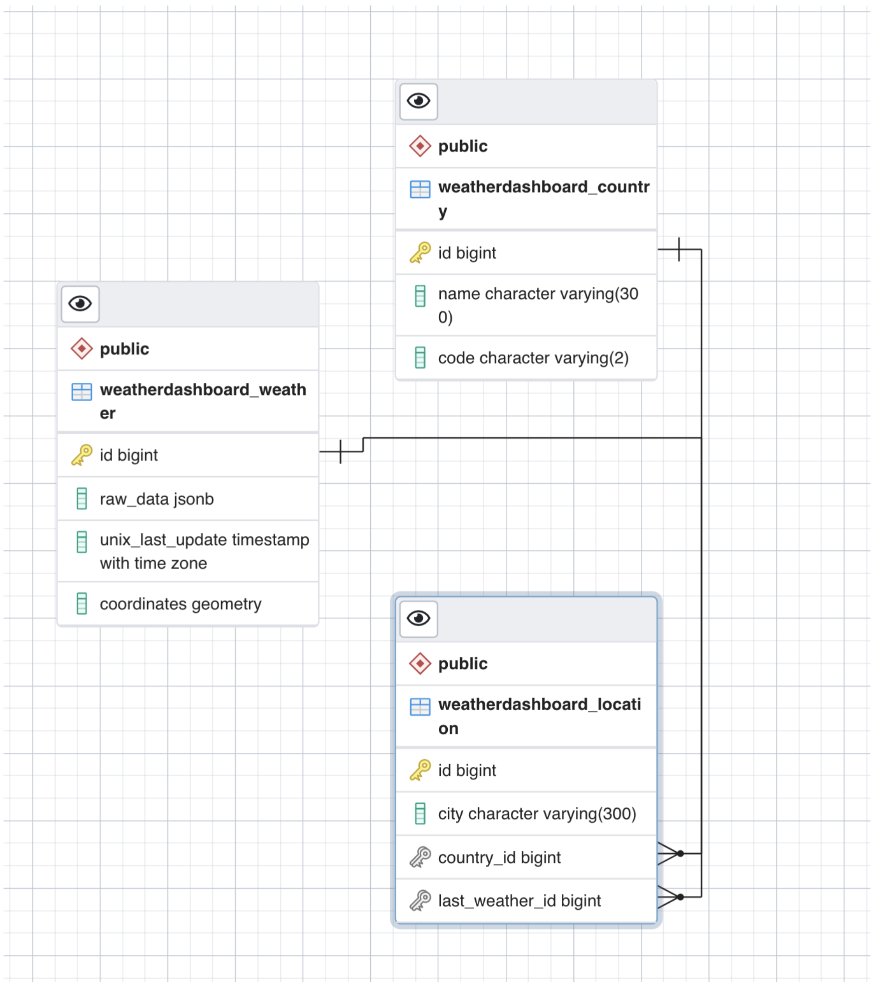
Fincite Challenge

frameworks used

- Data Base
    - Postgres
- Backend
    - Django
    - Django REST
- Frontend
    - React
    - Ant Design
- Weather Data
    - [openweathermap]()

ERD



- Country
    - stores all countries in the world
    - columns:
        - name: county name
        - code: country code
- Weather
    - stores all weather readings brought from weather API
    - columns:
        - raw_data: JSON returned by the weather API
        - unix_last_update: datetime of the weather reading
        - coordinates: coordinates of the weather reading
- Location
    - stores all locations (South American capitals by default) added by the user
    - columns:
        - city: name of the city
        - country: foreign key to country table
        - last_weather: foreign key to weather table

Backend

- APIs:
    - api/v1/
        - **countries**
            - lists all countries. Used in frontend for selecting a country in "add new location" feature
        - **locations**
            - lists all locations. Used in frontend for showing all locations at the bottom
        - **locations/add_location**
            - creates a new location. Used in frontend for creating a new location
                - if the location to be added already exists in database adds throws an exception
                - if the request body doesn't have all the request data, then throws an exception
                - body:
                    - dt: current datetime in unix timestamp
                    - city: city name
                    - country_code: country code
        - **locations/update_weather:**
            - returns all updated weathers of stored locations
            - params:
                - dt: current datetime in unix timestamp

            - if a stored location last weather has the same unix timestamp then returns that object, otherwise makes a request to the weather API, creates a new Weather register and returns it.
        - **weathers:**
            - **weathers/current_weather:**
                - returns weather of user current location
                - params:
                    - dt: current datetime in unix timestamp
                    - lat: current latitude
                    - lon: current longitude
                - if there is a register in a lat, lon with 20m distance and timestamp like the params then returns it, otherwise returns data from weather API
            - **weathers/forecast_weather:**
                - returns a list of the next 5 days weathers of user current location
                - that information could change every minute because that is not an approximate data like current weather, that's why those weather won't be stored in the database.
- Environment variables
    - NAME: name of the database
    - USER: database user
    - PASSWORD: database password
    - HOST: database host
    - PORT: database port
    - WEATHER_API_KEY: weather API key
    - LOCATION_BY_PLACE_API: location by place url from weather API
    - LOCATION_BY_COORD_API: location by coords url from weather API
    - FORECAST_BY_COORD_API: forecast by coords url from weather API

Frontend

- App.js
    - After loading:
        - executes every minute a function which makes:
            - current position request
            - current position weather request
            - forecast weather request
            - locations weather request
    - components:
        - CurrentPosition
            - shows current position data
            - components:
                - Loading: skeleton component while getting API data
                - WeatherForecast:

            - shows the next 5 days forecast data
            - components:
                - Loading: skeleton component while getting API data
- LocationsList
    - shows all locations weather data divided by 4 sections according to location temperature
    - adds a button and a modal for adding new locations
        - makes a validation if user is trying to add an existing location
    - components:
        - Loading: skeleton component while getting API data