

Weather Forecast for Macedonian Cities

This web application provides weather forecasts for Macedonian cities using Java Spring Boot on the back-end and Bootstrap-enabled HTML page on the front-end. It fetches weather data from the [OpenWeatherMap API](#), stores it in [PostgreSQL](#) database and allows users to view all forecasts or filter only hot weather days (where the max temperature exceeds 25°C).

➤ Project Structure

✓ Entity Layer

- **WeatherForecast.java**
 - Represents a weather forecast record with fields for city, date, max temperature and feels-like temperature. Annotated with [@Entity](#) for JPA persistence.

✓ Repository Layer

- **WeatherForecastRepository.java**
 - Extends [JpaRepository](#) to provide ORM access and defines custom query method:
[List<WeatherForecast> findByMaxTempGreaterThan\(double temp\);](#)

✓ Service Layer

- **WeatherForecastService.java** – Interface
- **WeatherForecastServiceImpl.java** – Implementation that implements core business logic:
 - [fetchAndStoreWeatherData\(\)](#) - fetches weather data from the OpenWeatherMap API and saves it to the database.
 - [FetchAllForecasts\(\)](#) – retrieves all forecast records.
 - [GetHotDays\(\)](#) – filters and returns forecasts where maxTemp > 25.0.

✓ Controller Layer

- **WeatherForecastController.java** that defines REST API endpoints:
 - [GET api/load](#) – Fetch and store weather data from the API.
 - [GET /api/all](#) – Retrieve all forecast data
 - [GET /api/hot-days](#) – Retrieve only hot weather days

✓ Frontend (HTML + Bootstrap + JavaScript)

- HTML page styled with Bootstrap 5
- Two buttons provided:
 - All Weather – loads all forecasts.
 - Hot Weather Only – filters and shows only hot days.
- Results are rendered in a styled table

- Uses **fetch()** to call the backend endpoints and dynamically display the results in the DOM.
- Accessible via:
<http://localhost:8081/weather.html>

✓ Database Configuration

- Database: PostgreSQL
- Connection Details (in application.properties):

```
spring.datasource.url=jdbc:postgresql://localhost:5432/weatherdb
spring.datasource.username=postgres
spring.datasource.password=postgres
```

- JPA Settings

```
# JPA settings
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
```

- API Integration

Uses the OpenWeatherMap API:

```
openweather.api.key = a4c0b3fe13e8f2deb1410da731f060f8
openweather.api.url=http://api.openweathermap.org/data/2.5/forecast
openweather.api.cities=Skopje:41.9981,21.4254;Gostivar:41.8000,20.9167;Ohrid:41.1231,20.8016
```