**Step-by-step guide** to creating a **Spring Boot Weather API** using **OpenWeatherMap API** in **Eclipse IDE**

**Step 1: Create a Free Account on OpenWeatherMap**

1. Go to OpenWeatherMap.
2. Sign up and log in.
3. Navigate to **API Keys** in your profile and copy your **API Key**.

**Step 2: Create a New Spring Boot Project in Eclipse**

1. **Open Eclipse IDE.**
2. **Go to File → New → Spring Starter Project.**
3. Enter the following details:
   * **Name:** WeatherApp
   * **Type:** Maven
   * **Packaging:** Jar
   * **Java Version:** 11 or later
   * **Group:** com.example
   * **Artifact:** weatherapp
4. Click **Next**, then select the following dependencies:
   * **Spring Web** (for REST APIs)
   * **Spring Boot DevTools** (for auto-restart during development)
5. Click **Finish.**

**Step 3: Add Dependencies in pom.xml**

Add the **WebClient** dependency for making HTTP requests.

In pom.xml, add this dependency inside <dependencies>:

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-webflux</artifactId>

</dependency>

<dependency>

<groupId>org.json</groupId>

<artifactId>json</artifactId>

<version>20231013</version>

</dependency>

**Step 4: Create the Weather Service**

1. Inside the **src/main/java/com/example/weatherapp** package:
   * **Right-click → New → Class**
   * Name it **WeatherService**
2. Add the following code in **WeatherService.java**:

*package com.example.weatherapp.service;*

*import org.springframework.stereotype.Service;*

*import org.springframework.web.reactive.function.client.WebClient;*

*import reactor.core.publisher.Mono;*

*@Service*

*public class WeatherService {*

*private static final String API\_KEY = "YOUR\_API\_KEY"; // Replace with your OpenWeatherMap API Key*

*private static final String BASE\_URL = "https://api.openweathermap.org/data/2.5/weather";*

*private final WebClient webClient;*

*public WeatherService(WebClient.Builder webClientBuilder) {*

*this.webClient = webClientBuilder.baseUrl(BASE\_URL).build();*

*}*

*public Mono<String> getWeather(String city) {*

*return webClient.get()*

*.uri(uriBuilder -> uriBuilder*

*.queryParam("q", city)*

*.queryParam("appid", API\_KEY)*

*.queryParam("units", "metric") // For Celsius*

*.build())*

*.retrieve()*

*.bodyToMono(String.class);*

*}*

*}*

**Step 5: Create the Weather Controller**

1. Inside the same package (com.example.weatherapp):
   * **Right-click → New → Class**
   * Name it **WeatherController**
2. Add the following code in **WeatherController.java**:

package com.example.weatherapp.controller;

import com.example.weatherapp.service.WeatherService;

import org.springframework.web.bind.annotation.\*;

import reactor.core.publisher.Mono;

@RestController

@RequestMapping("/api/weather")

public class WeatherController {

private final WeatherService weatherService;

public WeatherController(WeatherService weatherService) {

this.weatherService = weatherService;

}

@GetMapping

public Mono<String> getWeather(@RequestParam String city) {

return weatherService.getWeather(city);

}

}

**Step 6: Modify the Main Application Class**

Ensure your **WeatherAppApplication.java** looks like this:

package com.example.weatherapp;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class WeatherAppApplication {

public static void main(String[] args) {

SpringApplication.run(WeatherAppApplication.class, args);

}

}

**Step 7: Run the Application**

1. **Right-click the project → Run As → Spring Boot App.**

**Step 8: Test the API**

**Using Browser or Postman:**

* URL Example:

<http://localhost:8080/api/weather?city=Bangalore>

**Sample Response**

