Technical description

1. General description

DataStorage solution consists of four main packages:

- **com.example.weatherapi** – package for application, configuration, REST controller, and CLI runner. It is parent package for other packages
- **com.example.weatherapi.api** – classes for using external weather apis. Have logic for building and sending requests and got responses.
- **com.example.weatherapi.model** – interfaces and classes that models the solution of weather API
- **com.example.weatherapi.service** – interface and classes for modeled weather service

2. Projects details

Solution consists of the following elements:

1. **WeatherService**

   Defines logic for finding weather

   ```
   ResponseEntity<Weather>
         findCurrentWeather(City city);
   ```

   Classes that implements interface:

   `BaseWeatherService` – abstract class that handles RestClientExceptions

   `NinjasWeatherService` – class for getting wether by Ninjas Api

   `WeatherApiService` – class for getting weather by Weather Api

2. **Weather**

   Defines weather.

   ```
   Weather
         TemperatureName getTemperature();
         InsolationName getInsolation();
   ```

   Classes that implements interface:

`WeatherImpl`

3. **City**

   Defines city.

```
City<T>
      String getName();
      Optional<String> getPostalCode();
      Optional<String> getCountry();
```

   Classes that implements interface:

   `CityImpl`

4. **TemperatureName**

   Enum for temperature values: warm, cold

5. **InsolationName**

   Enum for insolation values: bright, cloudy

6. **TemperatureInterpreter**

   Defines logic for naming integer temperature value

```
TemperatureInterpreter
      TemperatureName interpret(double temperature);
```

   Classes that implements interface:

   `ThresholdTemperatureInterpreter` – simple interpreter based on one threshold value

7. **InsolationInterpreter**

   Defines logic for naming integer insolation value

```
InsolationInterpreter
      InsolationName interpret(int insolation);
```

   Classes that implements interface:

   `PercentageInsolationInterpreter` – simple interpreter based on one threshold value

8. **NinjasWeatherApi**

   Uses external Ninja Api to get weather. Uses RestTemplate for communication.

   Other classes used for communication:

`Response` – Defines response weather attributes

9. WeatherApi

Uses external Weather Api to get weather. Uses RestTemplate for communication.

Other classes used for communication:

`Response` – Defines response weather and location

`Location` – Defines response location

`WeatherAttributes` – Defines response weather attributes

10. Application

Spring Boot application

`WeatherApiApplication`

11. Controller

Defines web services that let to get current weather by http

`WeatherApiController`

12. Configuration

Defines Spring Beans

`WeatherApiConfiguration`

13. CommandLineRunner

Defines Command Line Runner for getting weather

`WeatherCommandLineRunner`