**Environmental Data API Documentation**

In this document explained about project flow and test scenarios.

**Project Setup:**

Initialize a new Spring Boot project using Spring Initializr.

Include dependencies for web, data-jpa, h2 (for in-memory database), and lombok (for reducing boilerplate code).

Set up your project structure, including packages for controller, service, repository, entity, exceptions and any other necessary components.

For this I have taken base package as com.enviro.assessment.gard001.seshagirisainni same as which mentioned in assessment document.

**Model Definition:**

Define the EnvironmentalData entity class to represent the data structure of environmental data.

Include fields for date, location, temperature, humidity, and airQuality.

Annotate the entity class with JPA annotations (@Entity, @Table, @Id, etc.) to map it to the database table.

**Repository Layer:**

Create a repository interface, EnvironmentalDataRepository, extending JpaRepository<EnvironmentalData, Long>.

Define custom query methods if needed for data retrieval or manipulation.

**Service Layer:**

Created a service interface and classes. EnvironmentalDataService , EnvironmentalDataServiceImpl, to implement the business logic for uploading and retrieving environmental data.

Implement the uploadFile method to parse the uploaded file and save environmental data to the database.

Implement the retrieveData method to fetch environmental data from the repository.

Implement the retrieveData method to fetch environmental data using id from the repository.

**Controller Layer:**

Created REST controller, EnvironmentalDataController, to define the API endpoints for uploading and retrieving environmental data.

Define a POST endpoint to handle file upload (/upload) and a GET endpoint to retrieve data (/retrieve).

**Logging**:

Integrated logging using SLF4J and a logging implementation with Logback.

Add log statements at appropriate places in the code to log important events, errors, or debug information.

**ExceptionHandling**:

Created custom Exception Handling class NoEnvironmentalDataFoundException.class to provide meaningful information about the error.

**Test scenarios:**

Base URL

**http://localhost:8080 /environmental-data**

**Endpoints**

**Upload Text File:**

• Endpoint: POST **/upload**

• Description: Upload a text file containing environmental data for analysis.

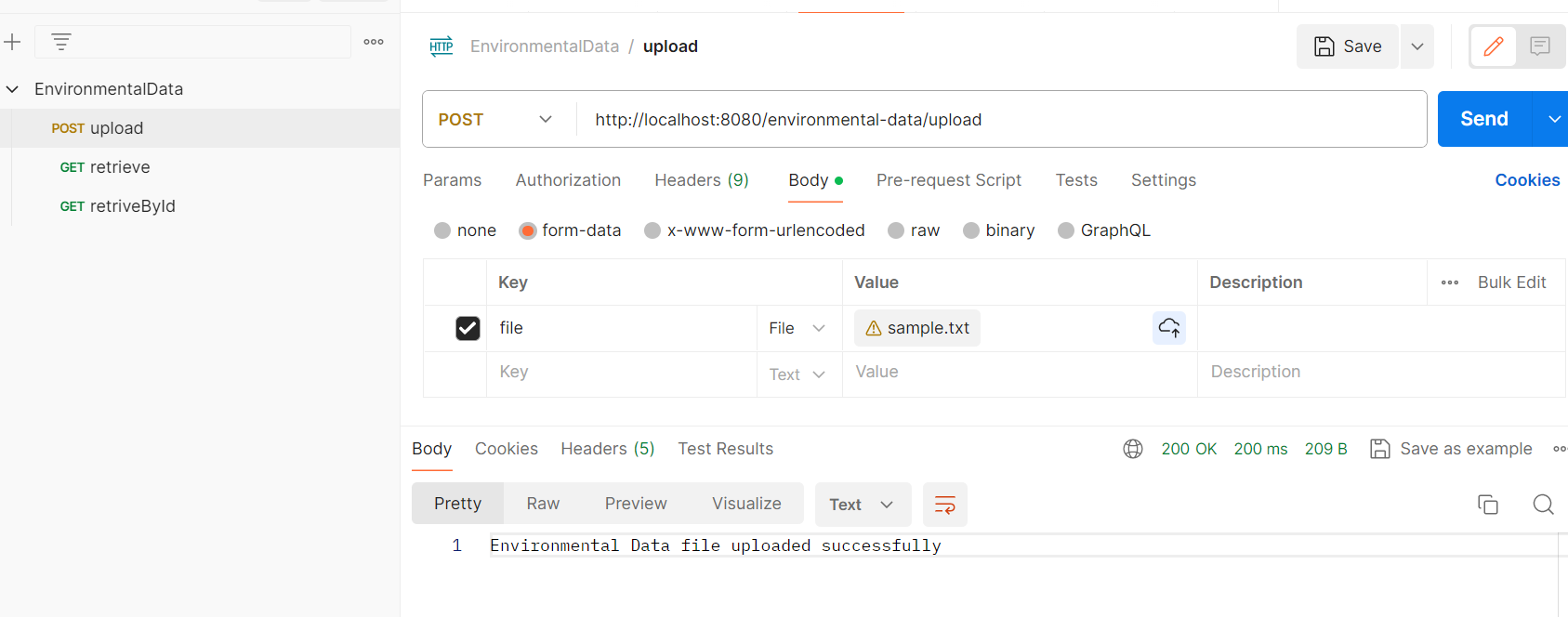
• Request Body: Select the file to upload in the body of the request. (ex: sample.txt)

• Response:

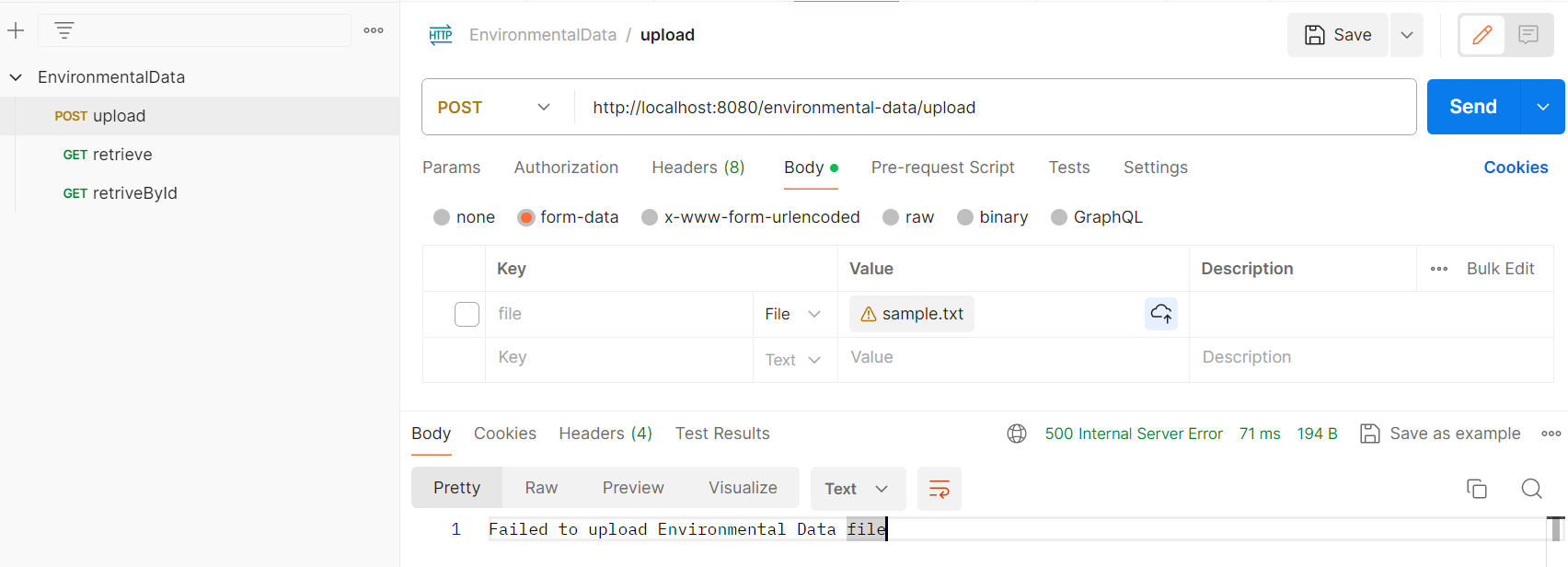
• Status: 200 OK - File uploaded successfully

• Status: 500 Internal Server Error - Failed to upload file (if an error occurs)

**Success scenario:**



**Failed scenario:**

****

**Retrieve Processed Data:**

• Endpoint: GET **/retrieve**

• Description: Retrieve processed environmental data from the database.

• Response:

• Status: 200 OK - Processed data retrieved successfully

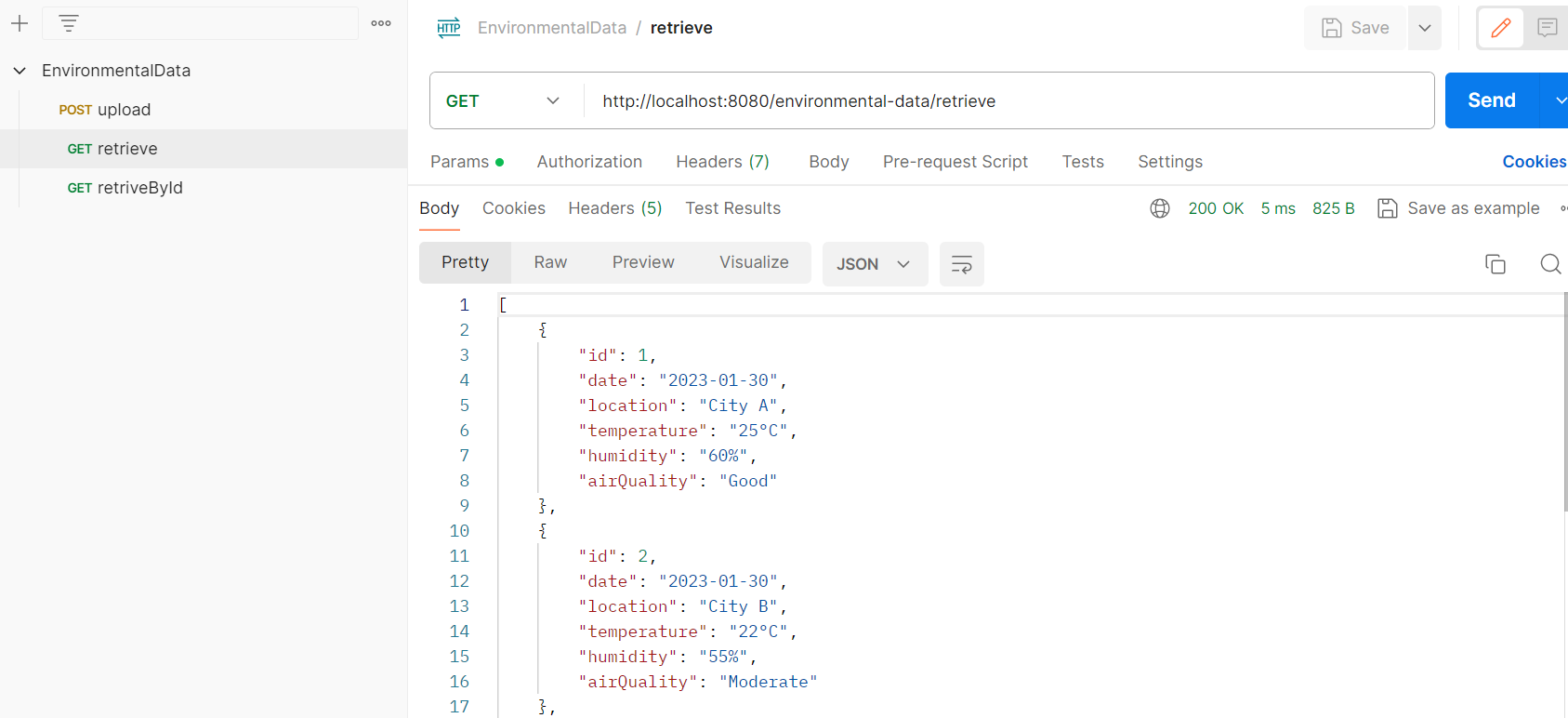
• Status: 400 Bad Request - No Environmental Data found

• Response Body:

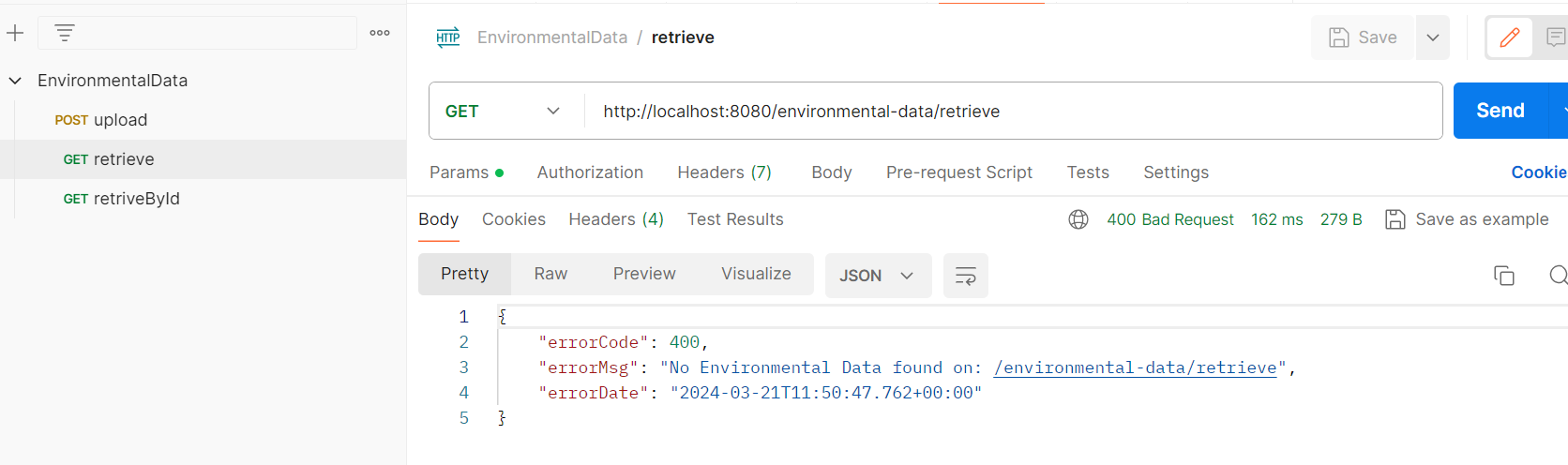
• Content-Type: text/plain

• Content: Processed environmental data retrieved from the database

**Success scenario:**

****

**Failed scenario:**



**Retrieve Processed Data by Id:**

• Endpoint: GET **/retrieve/{id}**

• Description: Retrieve processed environmental data from the database.

• Response:

• Status: 200 OK - Processed data retrieved successfully

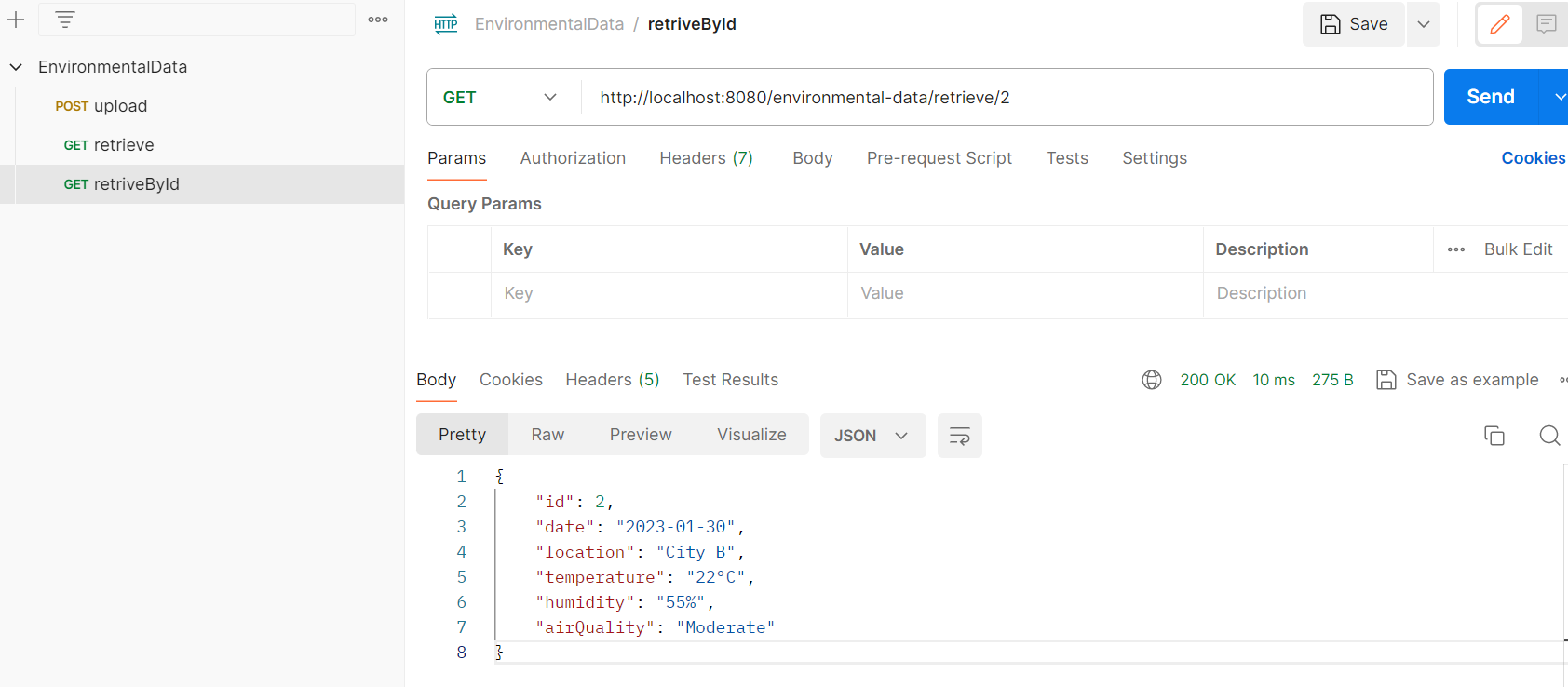
• Status: 400 Bad Request - No Environmental Data found

• Response Body:

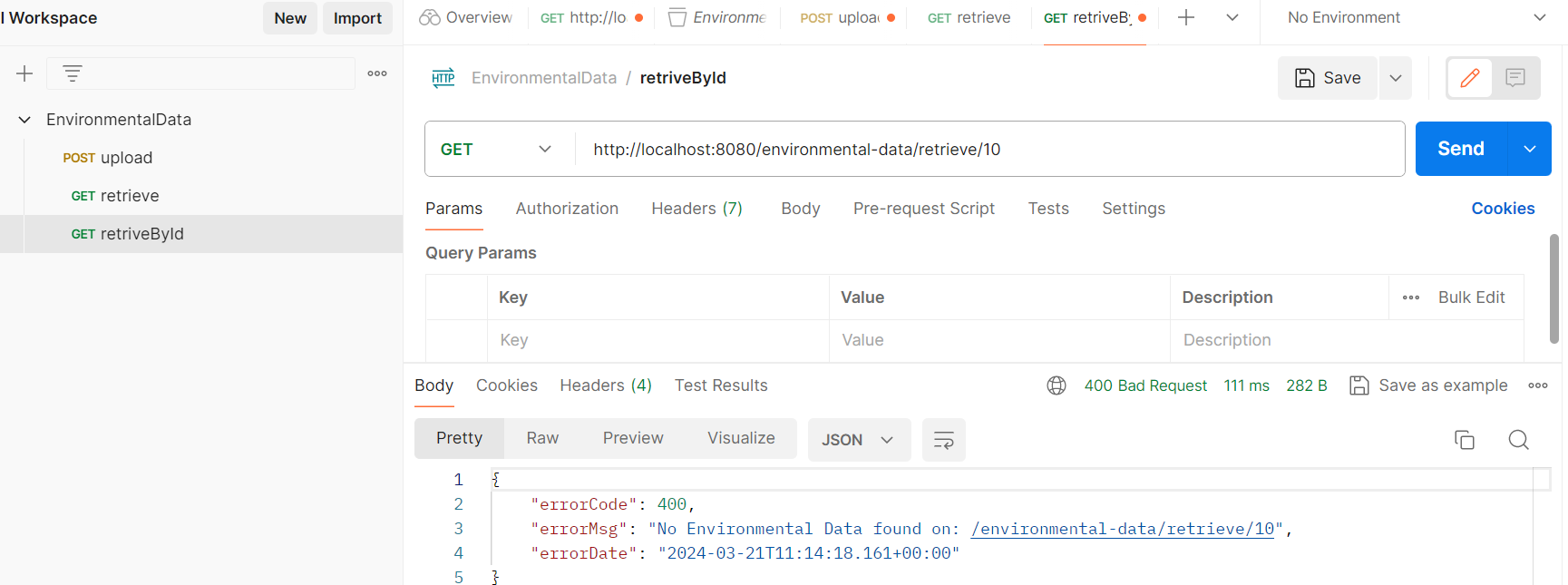
• Content-Type: text/plain

• Content: Processed environmental data retrieved from the database

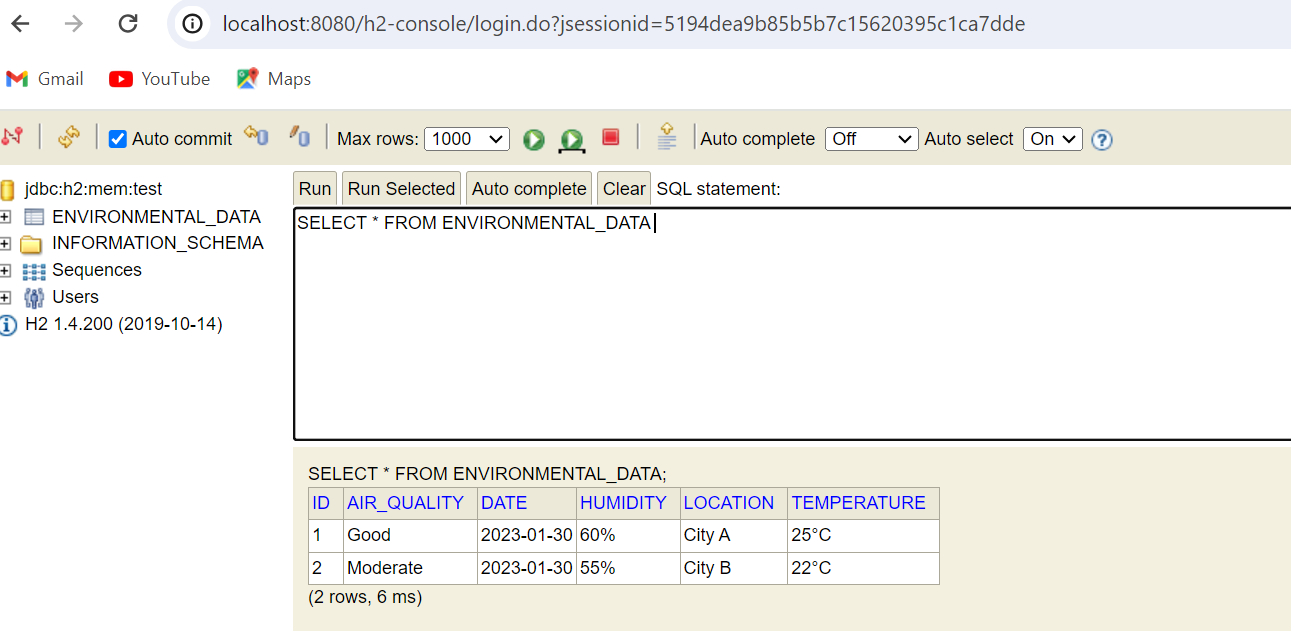
**Success scenario:**

****

**Failed scenario:**



**H2 Database console**



**Sample Usage**

Upload Text File

1. Open Postman and create a new POST request.

2. Set the request URL to http://localhost:8080/ environmental-data/upload

3. Select the form-data option for the request body.

4. Add a key-value pair with the key set to file and the value set to the text file containing environmental data.

5. Send the request.

Retrieve Processed Data

1. Open Postman and create a new GET request.

2. Set the request URL to http://localhost:8080/ environmental-data/retrieve

3. Send the request.

Retrieve Processed Data By Id

4. Open Postman and create a new GET request.

5. Set the request URL to http://localhost:8080/ environmental-data/retrieve/1

6. Send the request.

Notes

• Ensure that the Spring Boot application is running and accessible at http://localhost:8080 before sending requests.