Employee Project

Design document

This document contains information about the system design of the given exercise and instructions on executing it.

**Tools used:-**

1. IntelliJ editor for java programming.
2. SpringBoot framework for creating API.
3. MySQL server is used for creating given schemas.
4. Postman is used to make request to API.

**System Design:**

1. **Database:-**
   1. Created three tables (User, UserSummary and UserAddress) in a database, with format given in schemas.
   2. Inserted values into it manually.
2. **REST API’s:**
   1. Created separate entity for each table to fetch data from the table correspondingly.
   2. **Entities** - User, UserSummary, UserAddress
   3. **Controller** :-
      1. UserController – controls user entity
      2. SummaryController – controls user summary entity
      3. AddressController – controls user address entity
   4. **Repository** :-
      1. UserRepository – interface which holds user entity methods.
      2. SummaryRepository – interface which holds user summary entity methods.
      3. AddressRepsitory – interface which holds user address entity methods.
   5. **Service** :-
      1. UserService – points to the user-service API
      2. SummaryService – this API points to user-service getUserSummary method
      3. AddressService – points to user-address-service API.
3. **Application Properties** :-
   1. Configured the file to connect the MySQL server, with username, password and jdbc driver.
   2. Please change the configurations(username, password) according to your MYSQL server
4. **Spring Intializer**:
   1. Visited <https://start.spring.io/> and with Spring Webservices, Spring data JPA and Spring MySQL, selected Java 18 and Jar, with maven project.

**Requirements to execute:**

1. Latest Java version installed in the system.
2. MYSQL version 8.0.29
3. Install postman version 9.24.2 to send the request to API.
4. Install JDBC driver to access mysql server from java program.

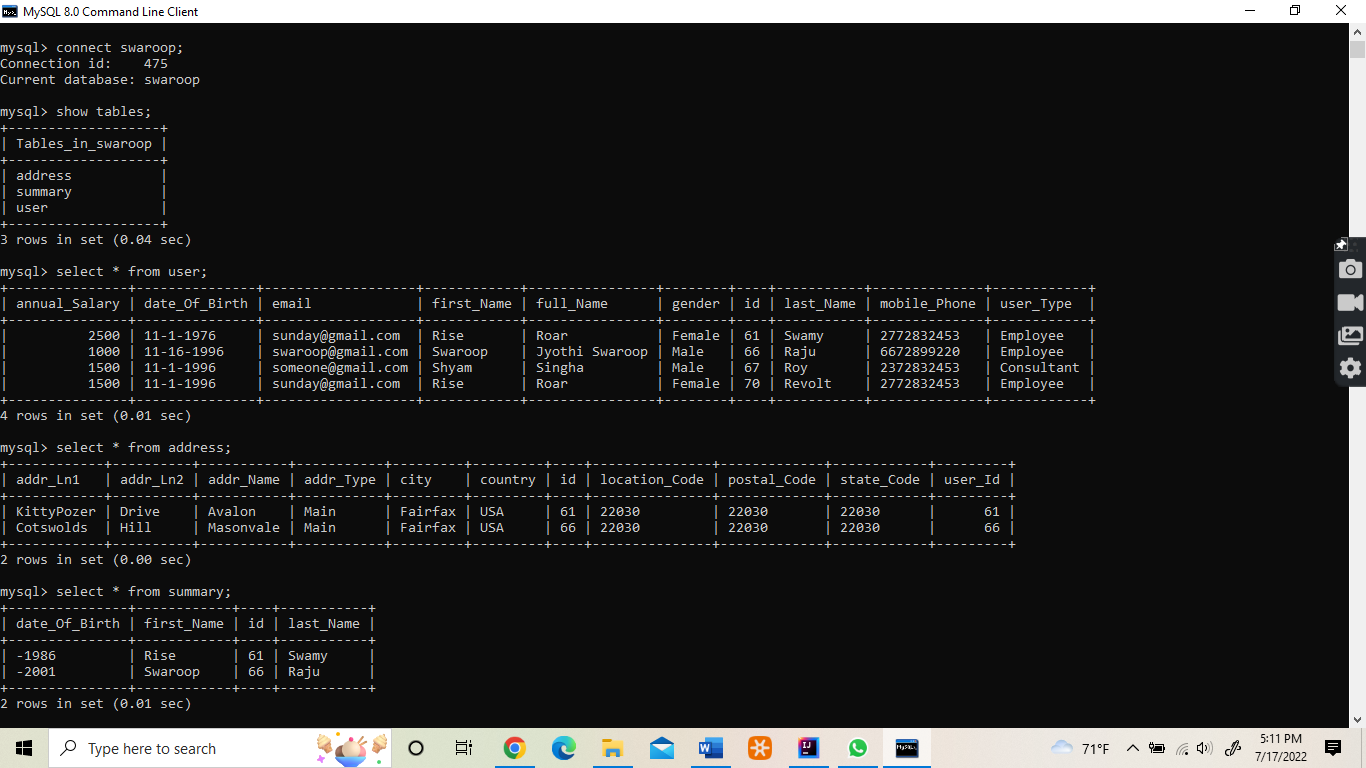
**Instruction to execute:-**

**MySQL Server side:**

1. Download the project from the Github.
2. Log in to your MySQL server, and execute the following commands.
   1. create database swaroop;
   2. connect swaroop;
   3. create table user (annual\_Salary int(10), date\_Of\_Birth varchar(10), email varchar(20),first\_Name varchar(24), full\_Name varchar(24), gender enum('Female','Male','Other'), id int(5), last\_Name varchar(24), mobile\_Phone varchar(10), user\_Type enum('Consultant','Employee'));
   4. alter table user add primary key(id);
   5. create table address (addr\_Ln1 varchar(10), addr\_Ln2 varchar(10), addr\_Name varchar(10), addr\_Type enum('Billing','Main','Shipping'), city varchar(10), country varchar(15), id int(5), location\_Code varchar(6), postal\_Code varchar(6), state\_Code varchar(6), userId int(5));
   6. alter table address add primary key(id);
   7. create table summary (date\_Of\_Birth varchar(10), first\_Name varchar(24), id int(5), last\_Name varchar(24));
   8. alter table summary add primary key(id);

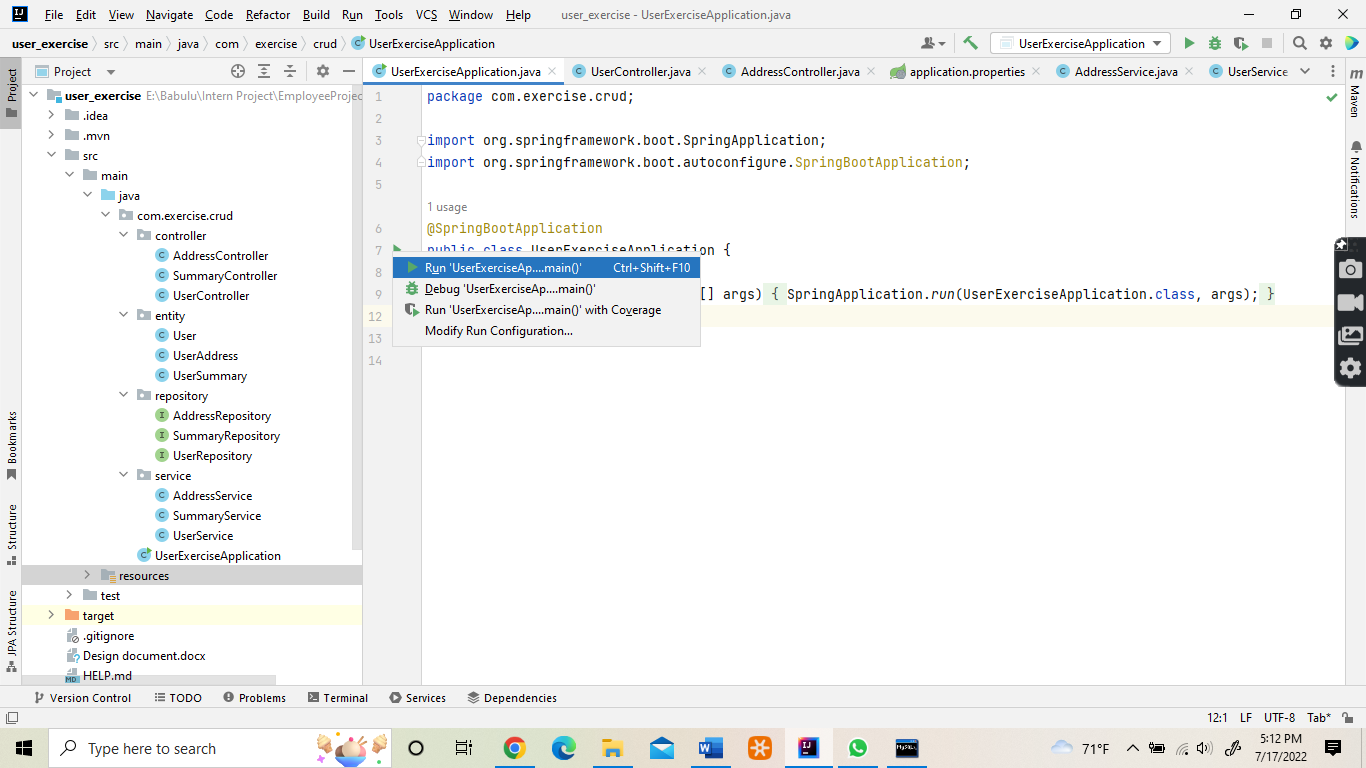
Now we have created the schemas for all the three entities. Insert few values into all these tables.

After adding some values into the table the database looks like this.

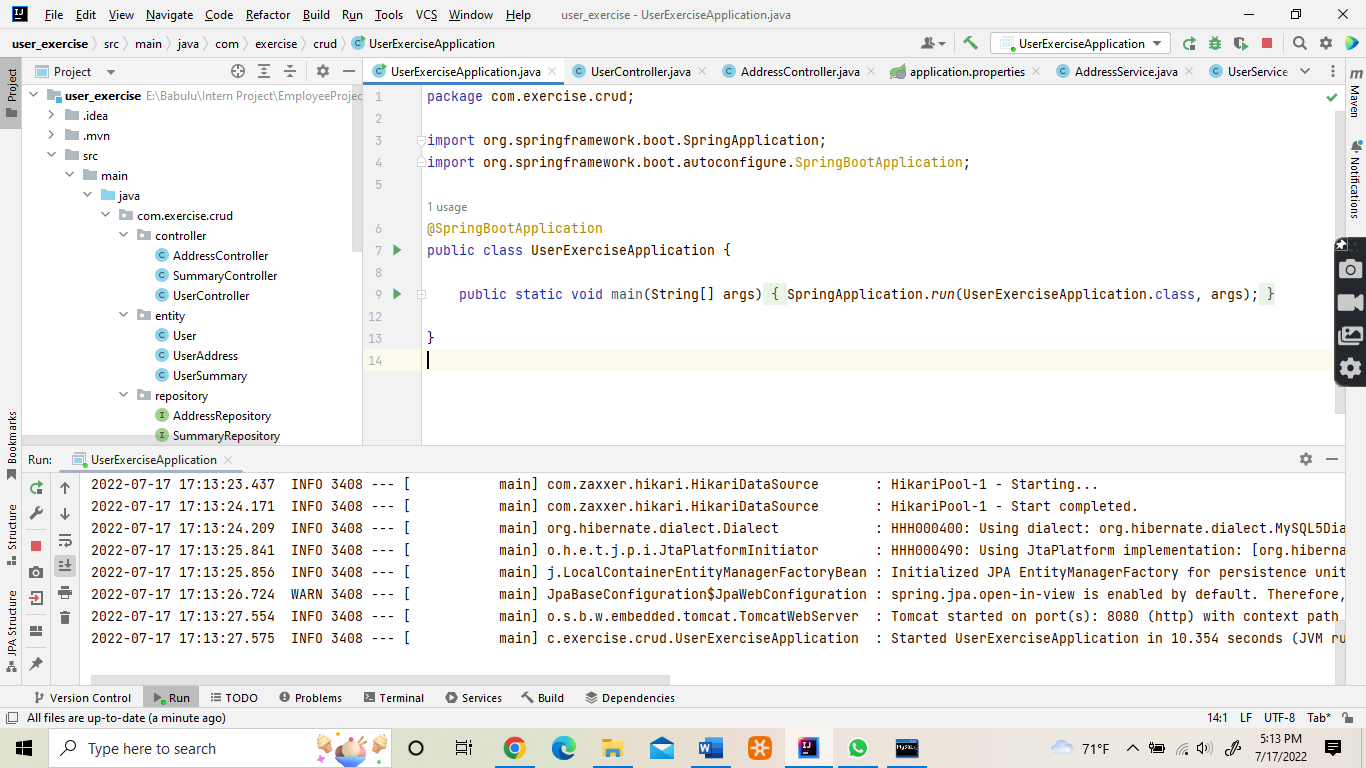


**Java program side:**

1. Open the project folder in the IntelliJ EmployeeProject/user\_exercise/src/main/java/com/exercise/crud.
2. Below is the snapshot of the project opened in IntelliJ, the UserExerciseApplication is the main application to run the program.



Running the UserExerciseApplication will look like below



Open the Postman application to validate the API

* User-service API:
  1. **POST** - localhost:8080/users
     1. To validate this API, copy paste the following sample JSON in the Body, of the raw option.

{

    "annualSalary": 25000,

    "dateOfBirth": "11-1-1976",

    "email": "project@gmail.com",

    "firstName": "firstName",

    "fullName": "fullName",

    "lastName": "lastName",

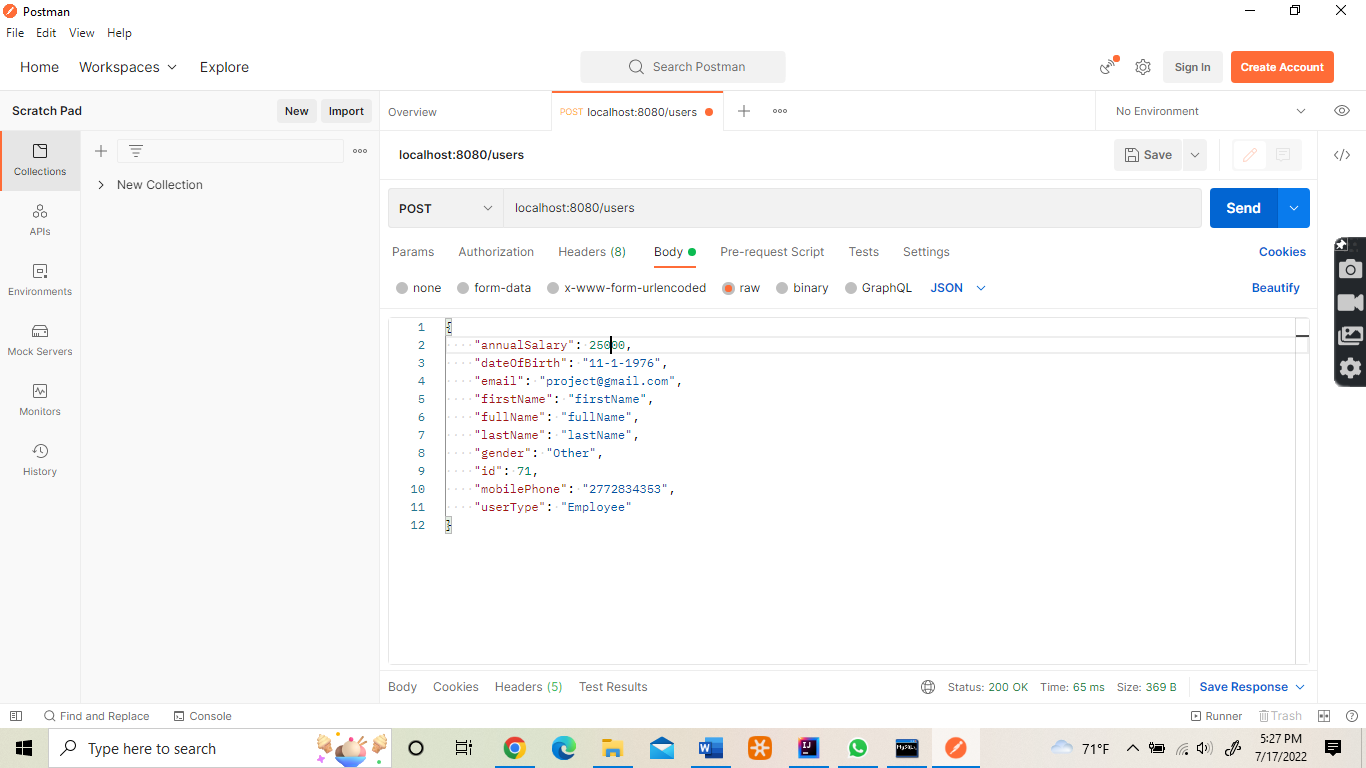
    "gender": "Other",

    "id": 71,

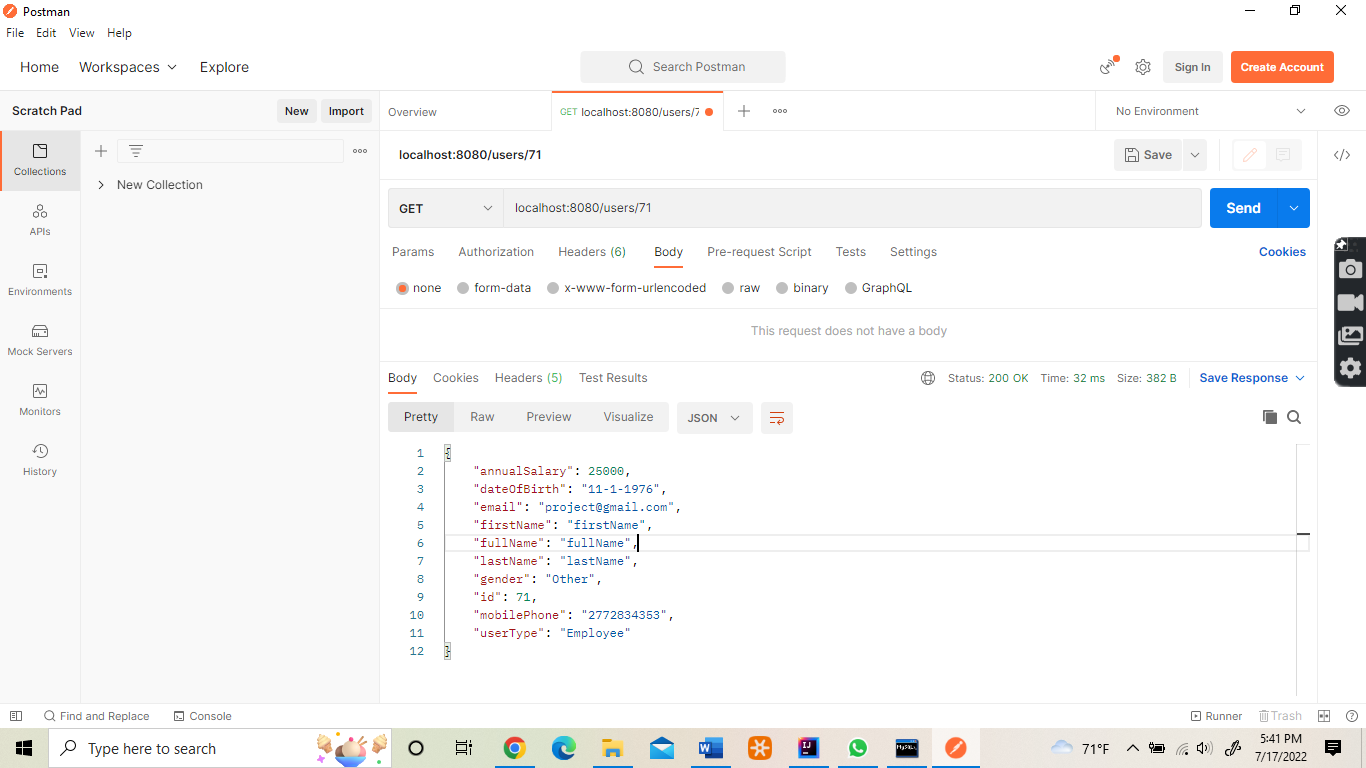
    "mobilePhone": "2772834353",

    "userType": "Employee"

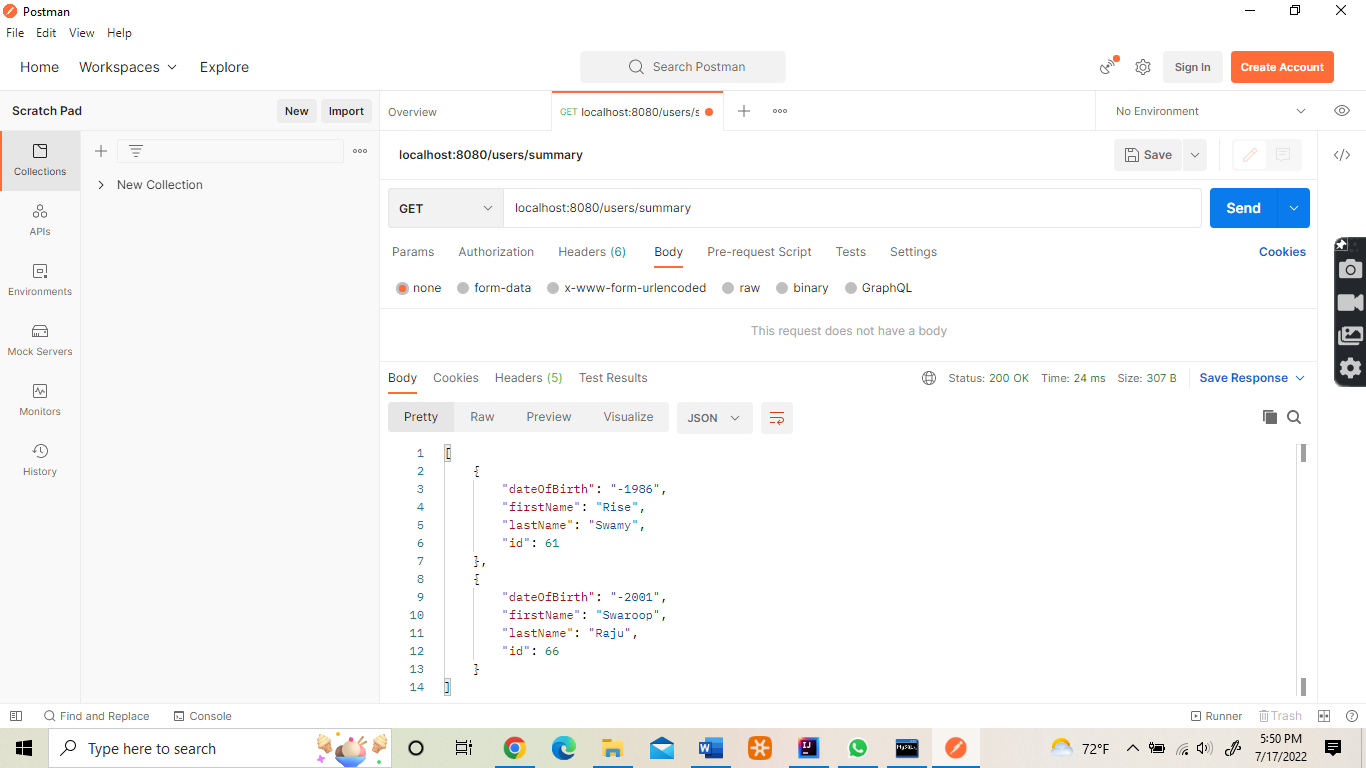
}



* 1. GET - localhost:8080/users/71
     1. Enter the above url in the Postman and click on Send button. Here the user id is 71, so it fetches the row of id 71.



* 1. GET - localhost:8080/users/summary
     1. This API fetches all the rows of summary entity, below is the snapshot of the output of the API.



* 1. PUT - localhost:8080/users/71
     1. This API updated the user id mentioned in path, here the user id is 71, content given in the body option in JSON format is taken as updated input. Paste the following command in the raw option of body part.

{

    "annualSalary": 25000,

    "dateOfBirth": "11-1-1906",

    "email": "intern@gmail.com",

    "firstName": "firstName",

    "fullName": "fullName",

    "lastName": "lastName",

    "gender": "Other",

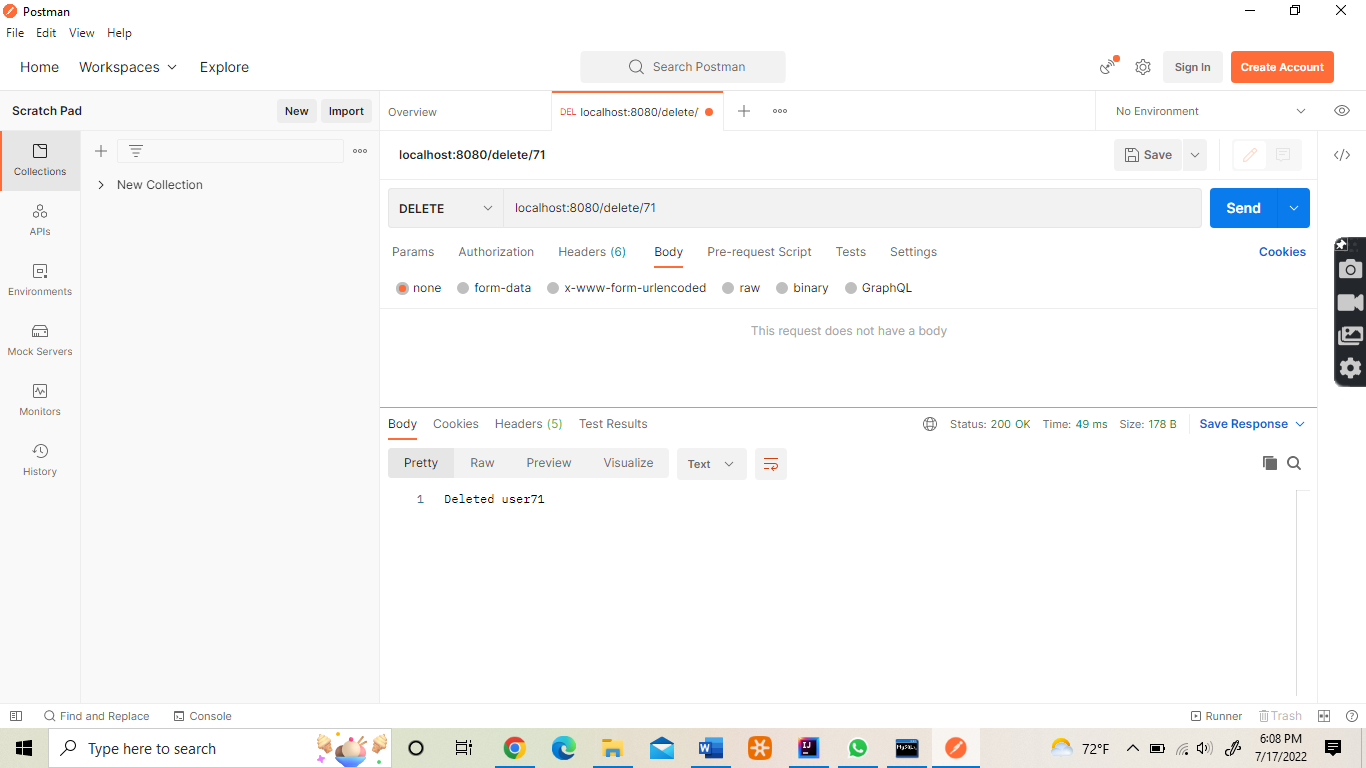
    "id": 71,

    "mobilePhone": "2772834353",

    "userType": "Employee"

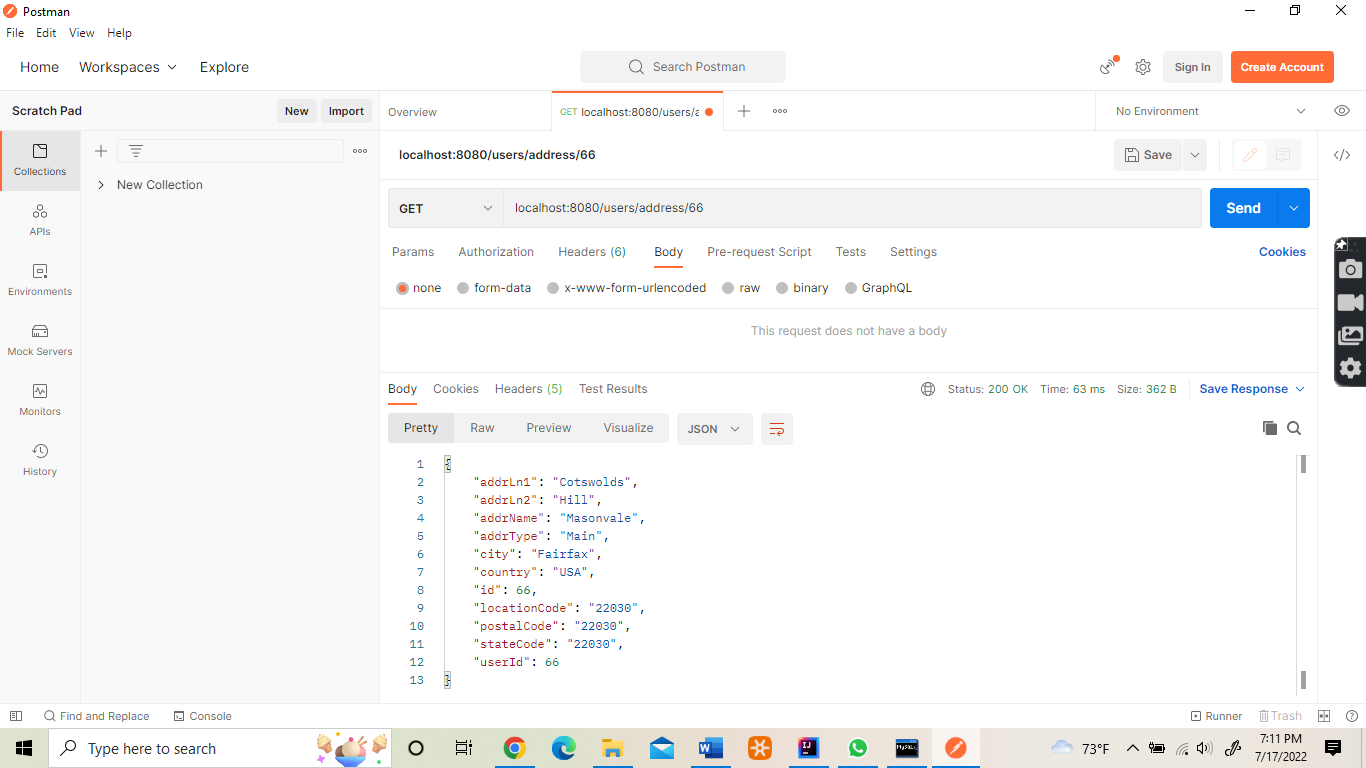
}

* 1. DELETE - localhost:8080/delete/71
     1. This API deletes the user ID mentioned in the path, and returns the message.



user- address – service API:

* GET - localhost:8080/users/address/66
  1. This API gets the user address of the id mentioned in the path.



* POST - localhost:8080/users/address
  1. This API saves the address to the mentioned used id in the body.
  2. Past the following JSON in the body part of the raw option,

{

    "addrLn1": "Cotswolds",

    "addrLn2": "Hill",

    "addrName": "Masonvale",

    "addrType": "Main",

    "city": "Fairfax",

    "country": "USA",

    "id": 66,

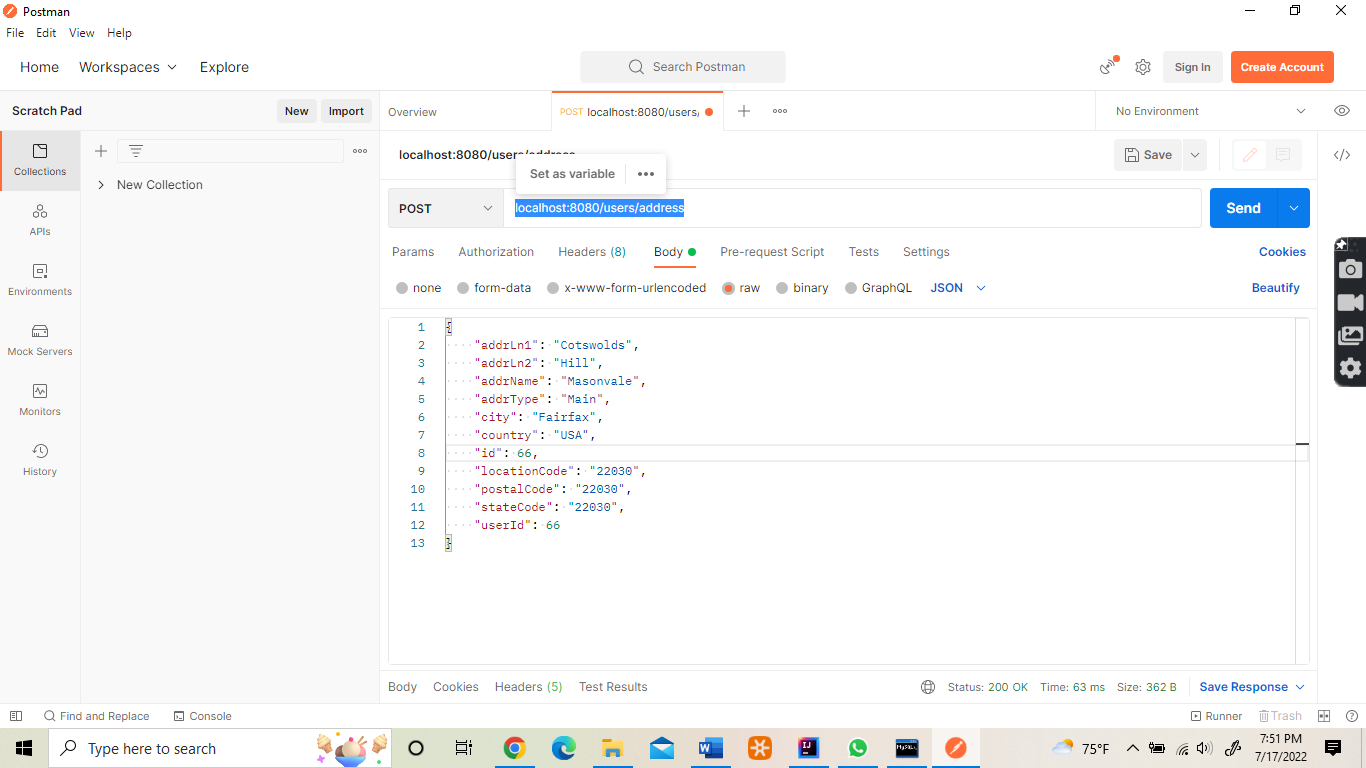
    "locationCode": "22030",

    "postalCode": "22030",

    "stateCode": "22030",

    "userId": 66

}



**Validations done:**

* User entity attributes were handled validation using proper annotations such as @NotNull, @Size, and @Pattern for date validation. Provided proper message if any field/data is missing in input wherever required.

**Improvements to be made:**

* Option to search for a given employee name/type should be implemented, tried it by including the findByName method but was facing issue while calling the API from Postman.