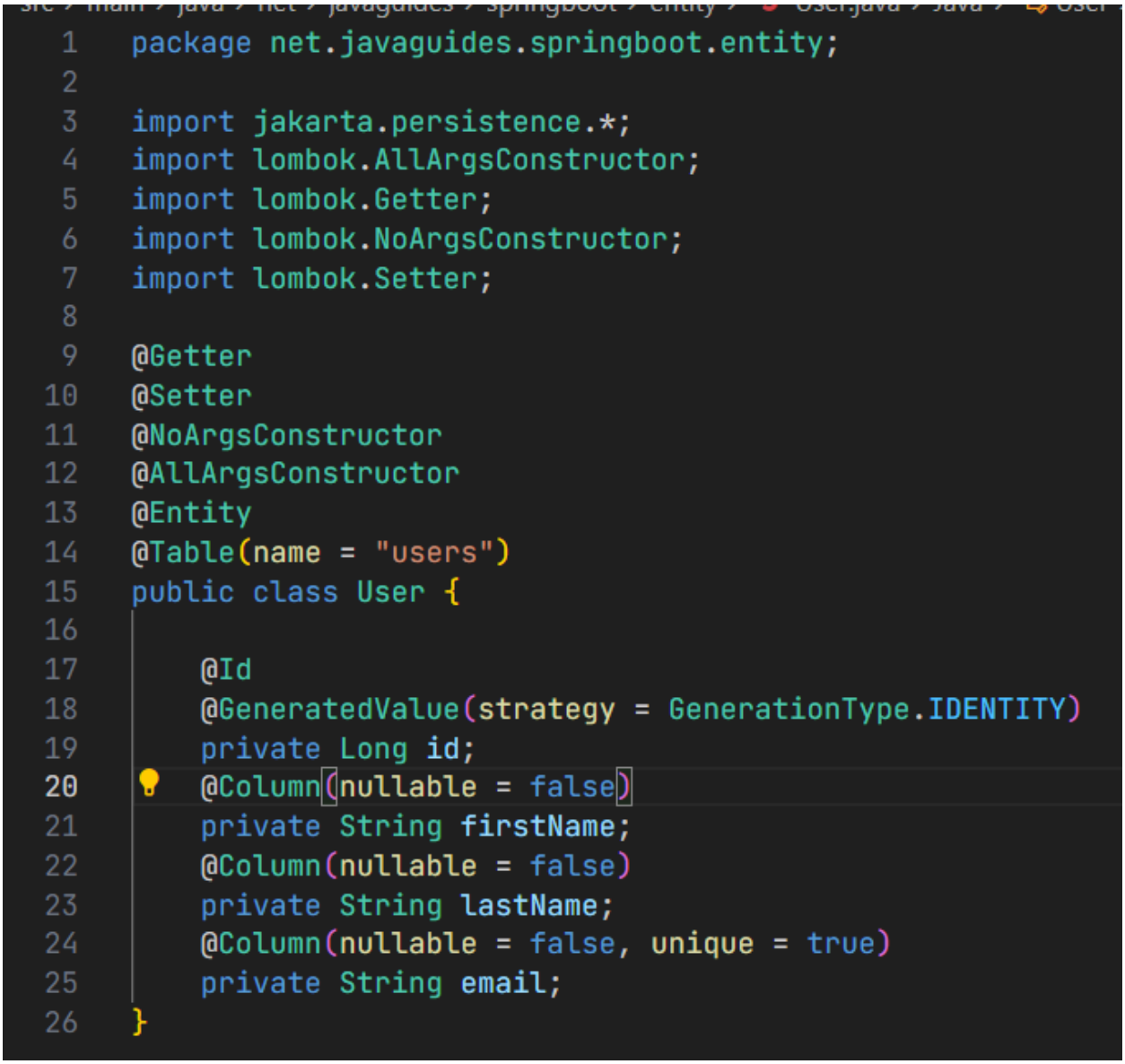
Fredluis A. Obidos

IV – BCSAD

# Assignment 1 - REST API

## Codes:

Create JPA Entity - User.java:



```java
package net.javaguides.springboot.entity;

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor
@Entity
@Table(name = "users")
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    @Column(nullable = false)
    private String firstName;
    @Column(nullable = false)
    private String lastName;
    @Column(nullable = false, unique = true)
    private String email;
}
```

Create Spring Data JPA Repository for User JPA Entity

```java
package net.javaguides.springboot.repository;

import net.javaguides.springboot.entity.User;
import org.springframework.data.jpa.repository.JpaRepository;

public interface UserRepository extends JpaRepository<User, Long> {
}
```

UserService Interface:

```java
package net.javaguides.springboot.service;

import net.javaguides.springboot.entity.User;

import java.util.List;

public interface UserService {
    User createUser(User user);

    User getUserById(Long userId);

    List<User> getAllUsers();

    User updateUser(User user);

    void deleteUser(Long userId);
}
```

UserServiceImpl Class:

```java
package net.javaguides.springboot.service.impl;

import lombok.AllArgsConstructor;
import net.javaguides.springboot.entity.User;
import net.javaguides.springboot.repository.UserRepository;
import net.javaguides.springboot.service.UserService;
import org.springframework.stereotype.Service;

import java.util.List;
import java.util.Optional;

@Service
@AllArgsConstructor
public class UserServiceImpl implements UserService {

    private UserRepository userRepository;

    @Override
    public User createUser(User user) {
        return userRepository.save(user);
    }

    @Override
    public User getUserById(Long userId) {
        Optional<User> optionalUser = userRepository.findById(userId);
        return optionalUser.get();
    }

    @Override
    public List<User> getAllUsers() {
        return userRepository.findAll();
    }

    @Override
    public User updateUser(User user) {
        User existingUser = userRepository.findById(user.getId()).get();
        existingUser.setFirstName(user.getFirstName());
        existingUser.setLastName(user.getLastName());
        existingUser.setEmail(user.getEmail());
        User updatedUser = userRepository.save(existingUser);
        return updatedUser;
    }
```

```java
    @Override
    public void deleteUser(Long userId) {
        userRepository.deleteById(userId);
    }
}
```

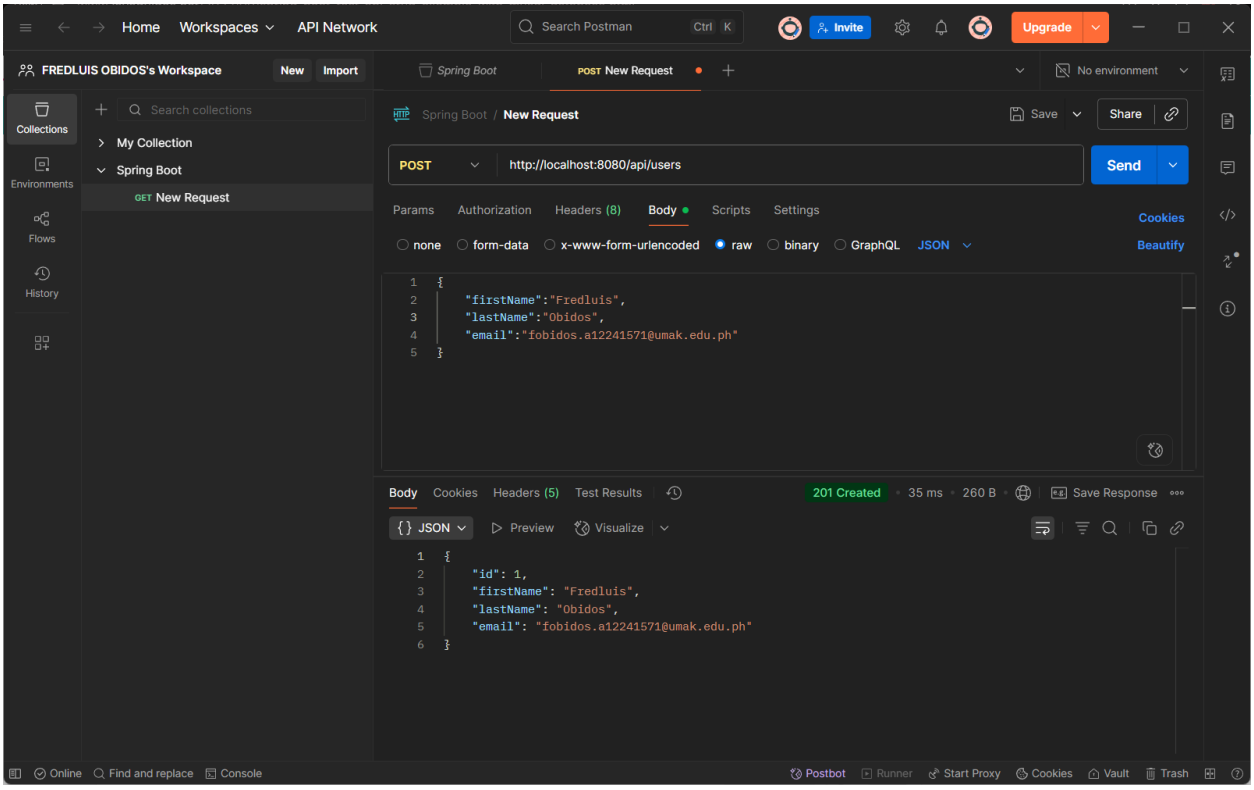
Creating UserController - Building CRUD Rest APIs:

```java
package net.javaguides.springboot.controller;

import lombok.AllArgsConstructor;
import net.javaguides.springboot.entity.User;
import net.javaguides.springboot.service.UserService;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
@AllArgsConstructor
@RequestMapping("api/users")
public class UserController {

    private UserService userService;

    // build create User REST API
    @PostMapping
    public ResponseEntity<User> createUser(@RequestBody User user){
        User savedUser = userService.createUser(user);
        return new ResponseEntity<>(savedUser, HttpStatus.CREATED);
    }

    // build get user by id REST API
    // http://localhost:8080/api/users/1
    @GetMapping("{id}")
    public ResponseEntity<User> getUserById(@PathVariable("id") Long userId){
        User user = userService.getUserById(userId);
        return new ResponseEntity<>(user, HttpStatus.OK);
    }
```

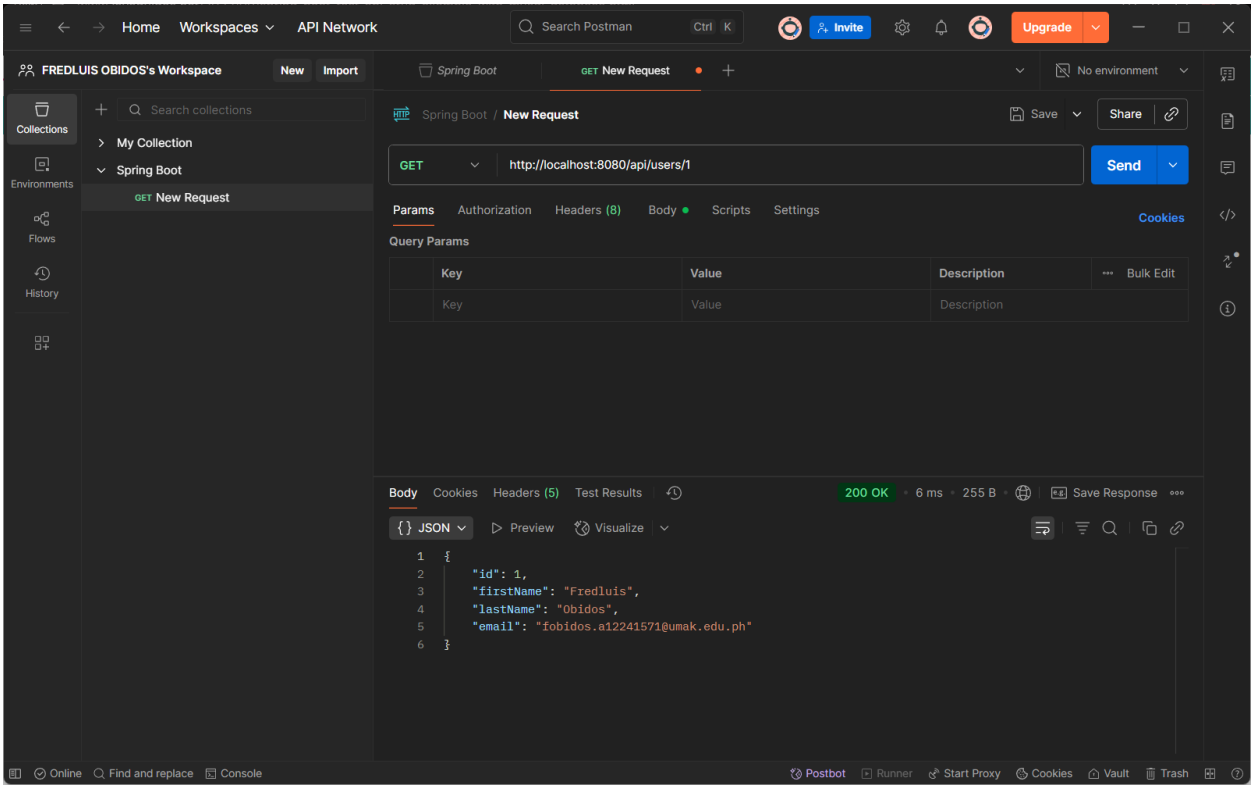
```java
    // Build Get All Users REST API
    // http://localhost:8080/api/users
    @GetMapping
    public ResponseEntity<List<User>> getAllUsers(){
        List<User> users = userService.getAllUsers();
        return new ResponseEntity<>(users, HttpStatus.OK);
    }

    // Build Update User REST API
    @PutMapping("{id}")
    // http://localhost:8080/api/users/1
    public ResponseEntity<User> updateUser(@PathVariable("id") Long userId,
                                           @RequestBody User user){
        user.setId(userId);
        User updatedUser = userService.updateUser(user);
        return new ResponseEntity<>(updatedUser, HttpStatus.OK);
    }

    // Build Delete User REST API
    @DeleteMapping("{id}")
    public ResponseEntity<String> deleteUser(@PathVariable("id") Long userId){
        userService.deleteUser(userId);
        return new ResponseEntity<>("User successfully deleted!", HttpStatus.OK);
    }
}
```

# Testing Screen Shots:

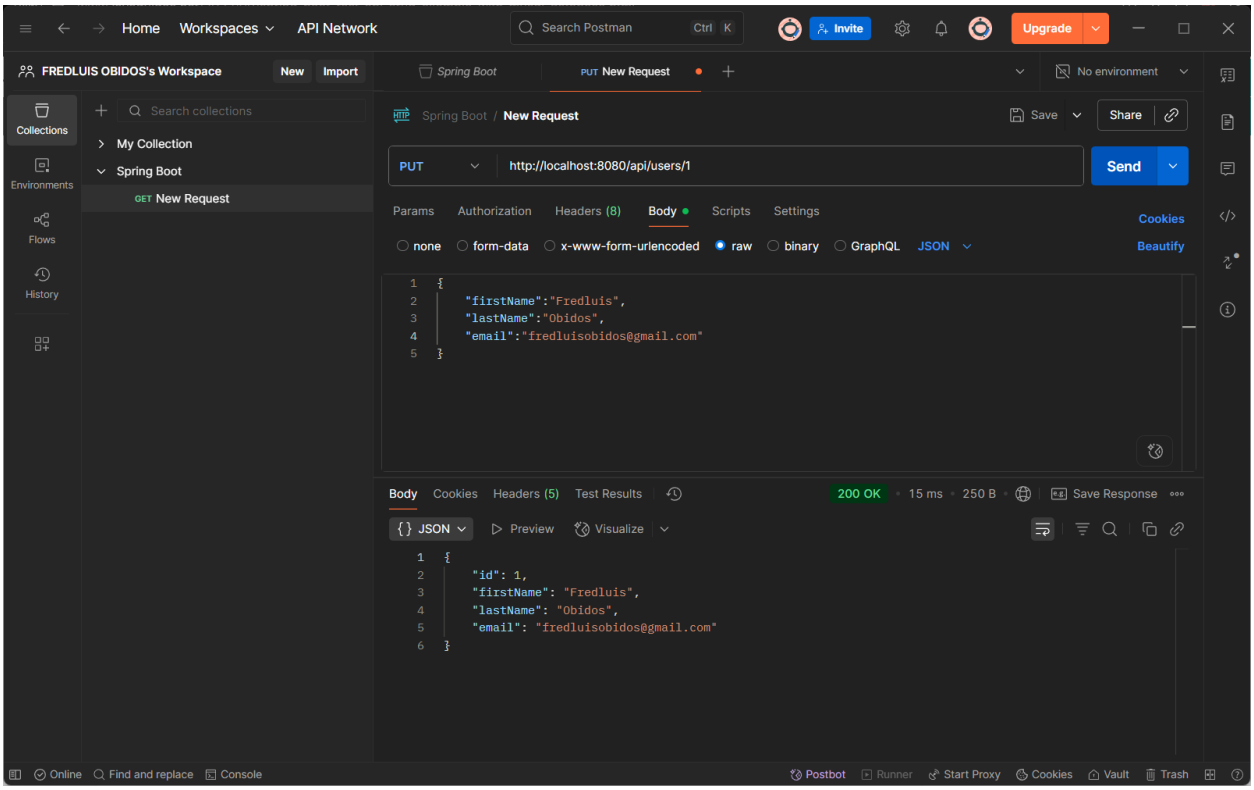
Create User REST API:



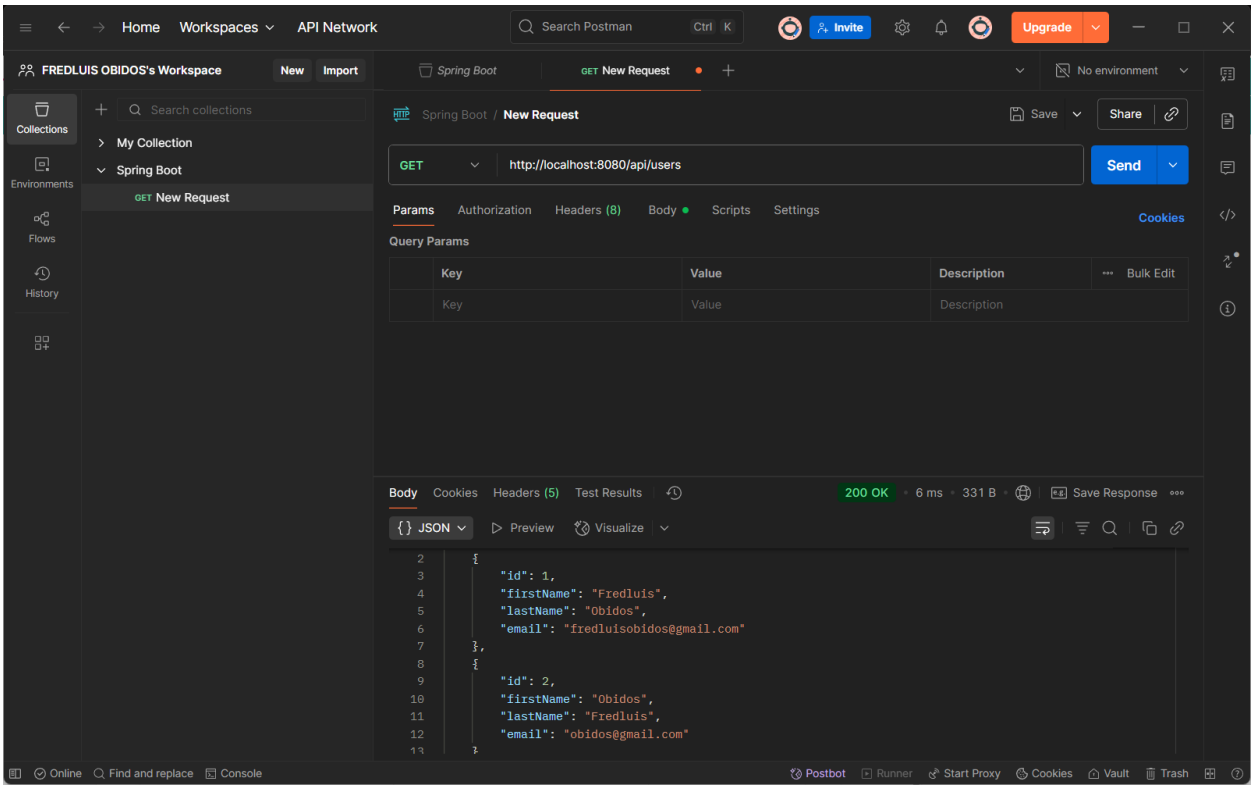
## Get Single User REST API:




## Update User REST API:



## Get All Users REST API:




## Delete User REST API: