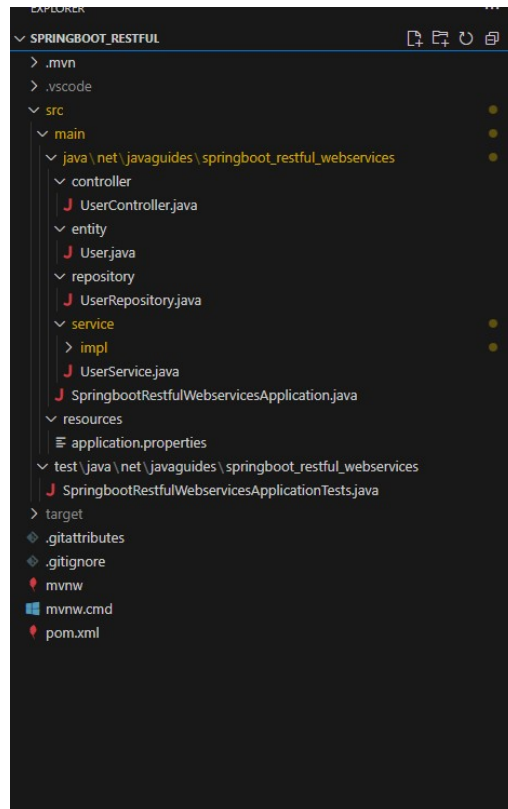


David Christian C. Olimberio

## IV – ACSAD

# Assignment 1 – REST API

## 1. Project Structure



## 2. MySQL Database

```
application.properties X
src > main > resources > application.properties
1  spring.application.name=springboot-restful-webservices
2  spring.datasource.url=jdbc:mysql://localhost:3306/user_management
3  spring.datasource.username=root
4  spring.datasource.password=
5
6  spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
7  spring.jpa.hibernate.ddl-auto=update
8
9  server.port=8080
```

### 3. JPA Entity – User. Java

```
J User.java
src > main > java > net > javaguides > springboot_restful_webservices > entity > J User.java > ...
1  package net.javaguides.springboot_restful_webservices.entity;
2  import jakarta.persistence.*;
3  import lombok.AllArgsConstructor;
4  import lombok.Getter;
5  import lombok.NoArgsConstructor;
6  import lombok.Setter;
7
8  @Getter
9  @Setter
10 @NoArgsConstructor
11 @AllArgsConstructor
12 @Entity
13 @Table(name = "users")
14 public class User {
15
16     @Id
17     @GeneratedValue(strategy = GenerationType.IDENTITY)
18     private Long id;
19     @Column(nullable = false)
20     private String firstName;
21     @Column(nullable = false)
22     private String lastName;
23     @Column(nullable = false, unique = true)
24     private String email;
25 }
26
```

### 4. Spring Data JPA Repository for User JPA Entity

```
J User.java  J UserRepository.java
src > main > java > net > javaguides > springboot_restful_webservices > repository > J UserRepository.java > UserRepository
1  package net.javaguides.springboot_restful_webservices.repository;
2
3  import net.javaguides.springboot_restful_webservices.entity.User;
4  import org.springframework.data.jpa.repository.JpaRepository;
5
6  public interface UserRepository extends JpaRepository<User, Long> {
7
8  }
9
```

## 5. Service Layer Implementation

```
21 public User createUser(User user) {
22     return userRepository.save(user);
23 }
24
25 @Override
26 public User getUserById(Long userId) {
27     Optional<User> optionalUser = userRepository.findById(userId);
28     return optionalUser.get();
29 }
30
31 @Override
32 public List<User> getAllUsers() {
33     return userRepository.findAll();
34 }
35
36 @Override
37 public User updateUser(User user) {
38     User existingUser = userRepository.findById(user.getId()).get();
39     existingUser.setFirstName(user.getFirstName());
40     existingUser.setLastName(user.getLastName());
41     existingUser.setEmail(user.getEmail());
42     User updatedUser = userRepository.save(existingUser);
43     return updatedUser;
44 }
45
46 @Override
47 public void deleteUser(Long userId) {
48     userRepository.deleteById(userId);
49 }
50 }
```

```
J User.java • J UserService.java X J UserRepository.java •
src > main > java > net > javaguides > springboot_restful_webservices > service > J UserService.java > •o UserServic
1 package net.javaguides.springboot_restful_webservices.service;
2
3 import net.javaguides.springboot_restful_webservices.entity.User;
4 import java.util.List;
5
6 public interface UserService {
7     User createUser(User user);
8
9     User getUserById(Long userId);
10
11     List<User> getAllUsers();
12
13     User updateUser(User user);
14
15     void deleteUser(Long userId);
16 }
```

## 6. UserController – CRUD Rest APIs

```
User.java • UserController.java • UserRepository.java •
src > main > java > net > javaguides > springboot_restful_webservices > controller > UserController.java > UserController

22 @RequestMapping("api/users")
23 public class UserController {
24
25     private UserService userService;
26
27     // Create User REST API
28     @PostMapping
29     public ResponseEntity<User> createUser(@RequestBody User user){
30         User savedUser = userService.createUser(user);
31         return new ResponseEntity<>(savedUser, HttpStatus.CREATED);
32     }
33
34     // Get user by id REST API
35     // http://localhost:8080/api/users/1
36     @GetMapping("{id}")
37     public ResponseEntity<User> getUserById(@PathVariable("id") Long userId){
38         User user = userService.getUserById(userId);
39         return new ResponseEntity<>(user, HttpStatus.OK);
40     }
41
42     // Get Users REST API
43     // http://localhost:8080/api/users
44     @GetMapping
45     public ResponseEntity<List<User>> getAllUsers(){
46         List<User> users = userService.getAllUsers();
47         return new ResponseEntity<>(users, HttpStatus.OK);
48     }
49
50     // Update User REST API
51     @PutMapping("{id}")
52     // http://localhost:8080/api/users/1
53     public ResponseEntity<User> updateUser(@PathVariable("id") Long userId,
54                                           @RequestBody User user){
55         user.setId(userId);
56         User updatedUser = userService.updateUser(user);
57         return new ResponseEntity<>(updatedUser, HttpStatus.OK);
58     }
59
60     // Delete User REST API
61     @DeleteMapping("{id}")
62     public ResponseEntity<String> deleteUser(@PathVariable("id") Long userId){
63         userService.deleteUser(userId);
64         return new ResponseEntity<>("User successfully deleted!", HttpStatus.OK);
65     }
66 }
```

## 7. Running the springboot

[illegible]



## 8. Creating the database using mysql

```
MariaDB [(none)]> create database user_management
-> ;
Query OK, 1 row affected (0.001 sec)

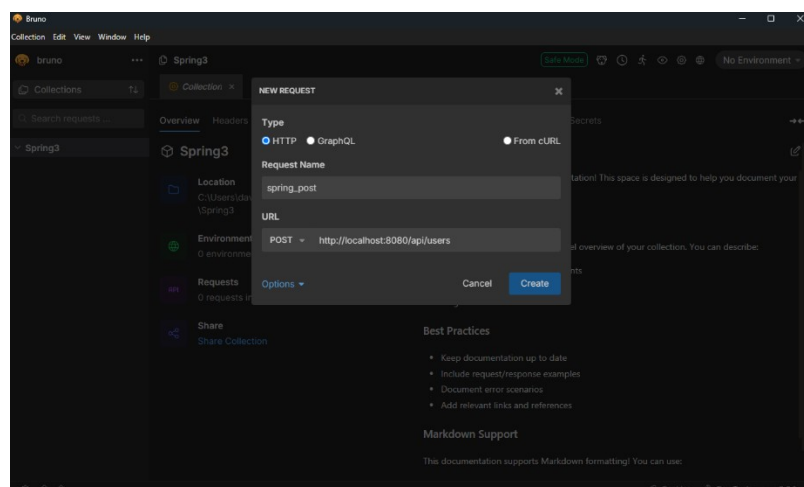
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| phpmyadmin |
| purple_hotel |
| test |
| user_management |
+-----+
7 rows in set (0.001 sec)

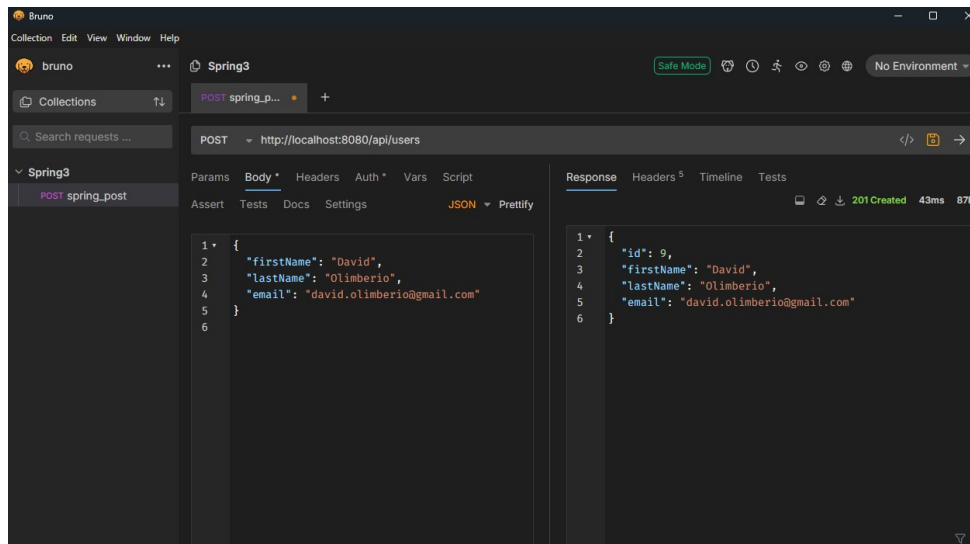
MariaDB [(none)]> USE user_management;
Database changed
MariaDB [user_management]> show tables;
+-----+
| Tables_in_user_management |
+-----+
| users |
+-----+
1 row in set (0.002 sec)

MariaDB [user_management]> |
```

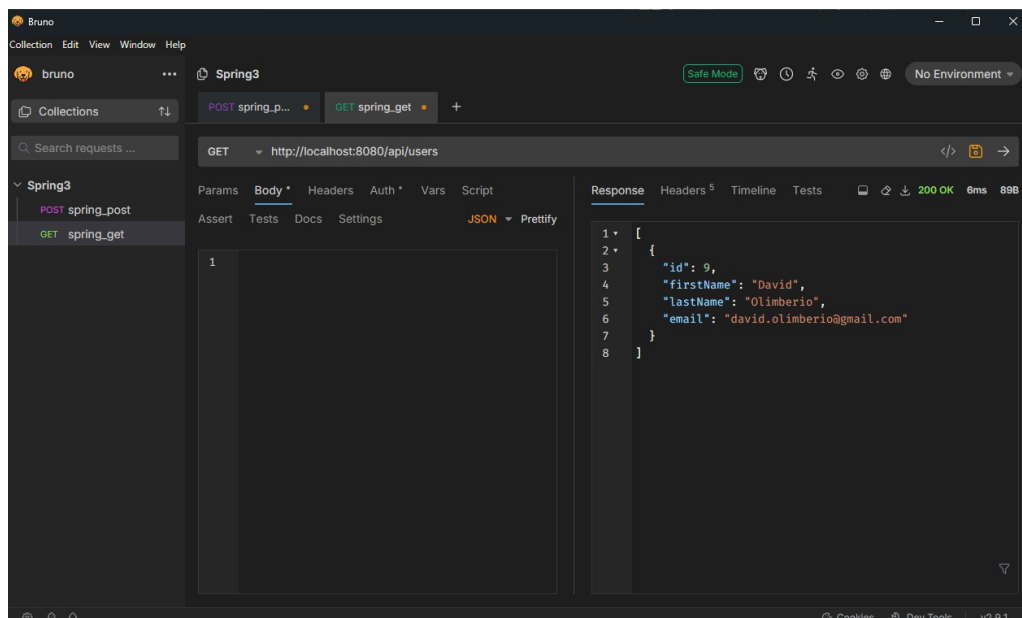
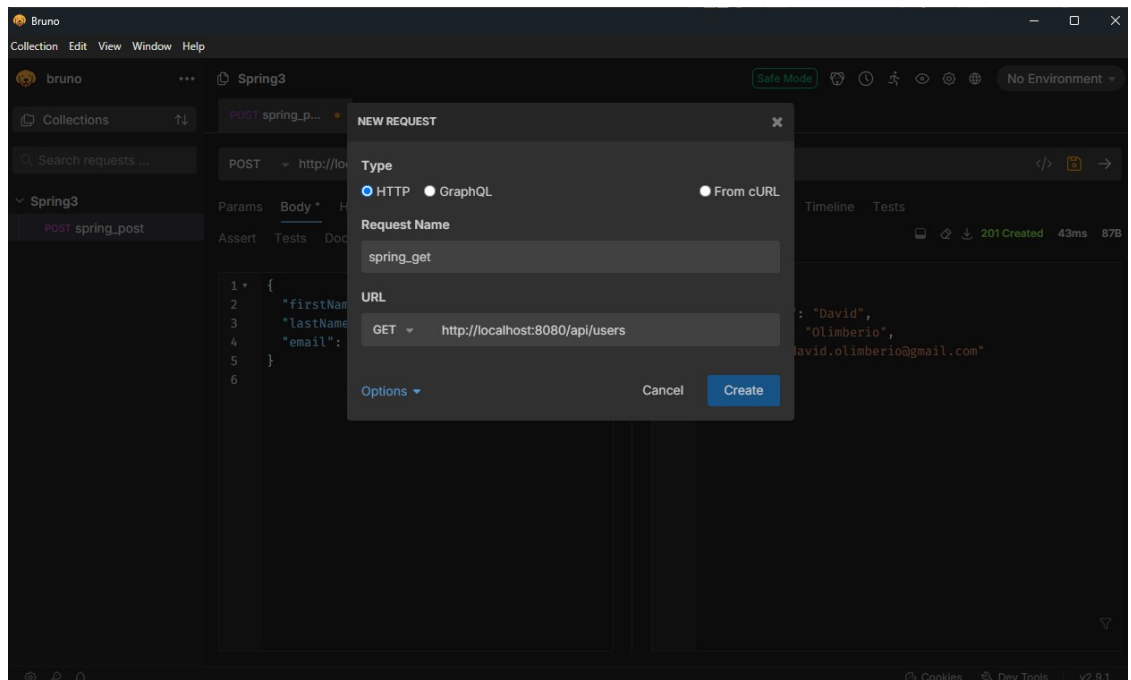
## 9. Testing with Bruno Rest API

### a. POST

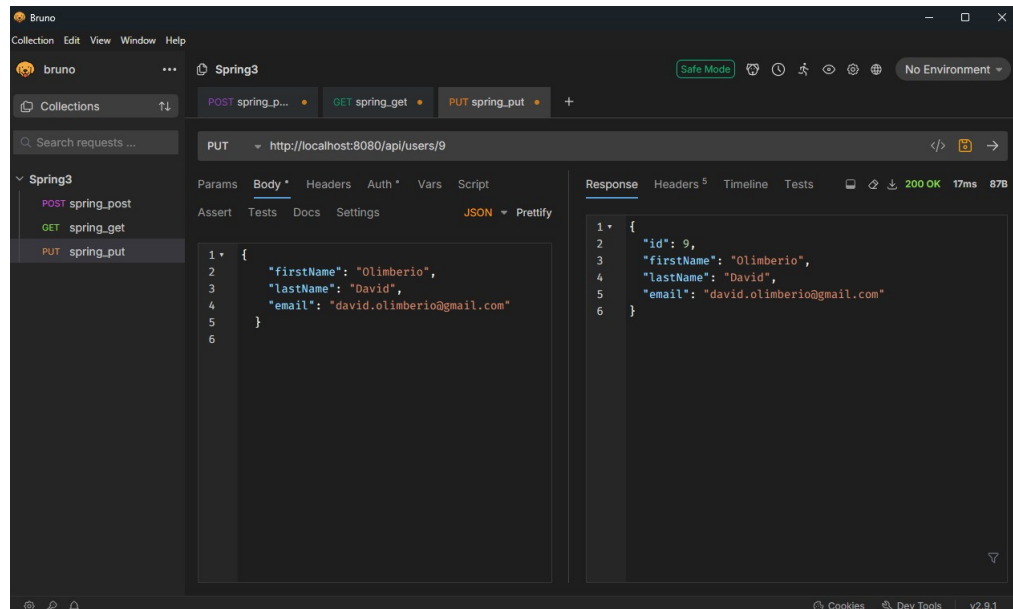
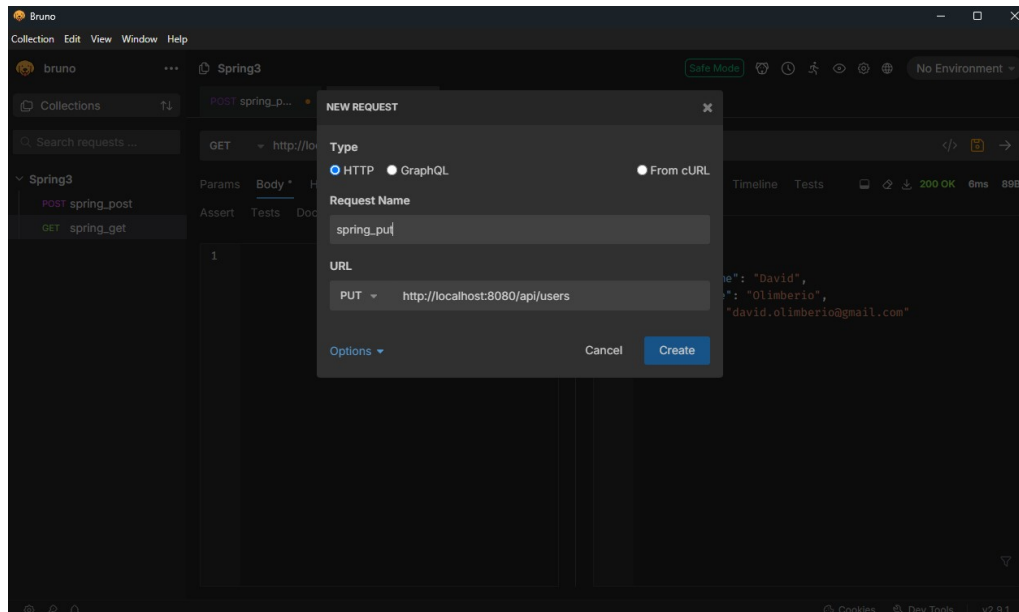




## b. GET



## c. PUT





## d. DELETE

