Sicat, Sean Russel

IV-ACSAD

REST API

Source Code:

1. **UserController.java**

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

import lombok.AllArgsConstructor;
import net.javaguides.springboottest.entity.User;
import net.javaguides.springboottest.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);
    }

    // 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);
    }
}
```

2. **User.java**

```java
package net.javaguides.springboottest.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;
}
```

3. **UserRepository.java**

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

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

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

4. UserServiceImpl.java

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

import lombok.AllArgsConstructor;
import net.javaguides.springboottest.entity.User;
import net.javaguides.springboottest.repository.UserRepository;
import net.javaguides.springboottest.service.UserService;
import org.apache.logging.log4j.util.Strings;
import org.springframework.stereotype.Service;
import org.springframework.util.StringUtils;

import java.util.List;
import java.util.Objects;
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;
    }

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

5. **UserService.java**

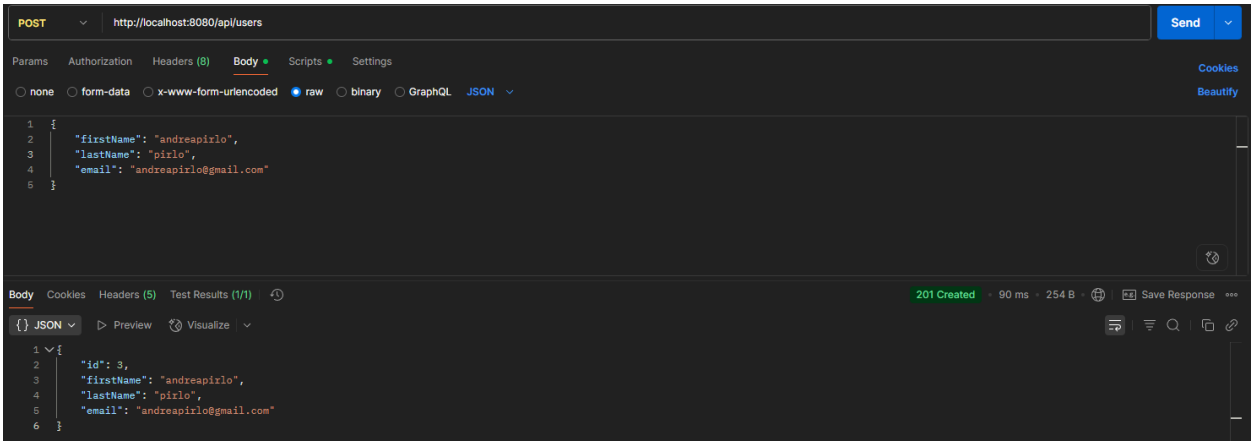
```java
package net.javaguides.springboottest.service;

import net.javaguides.springboottest.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);
}
```

6. **application.properties**

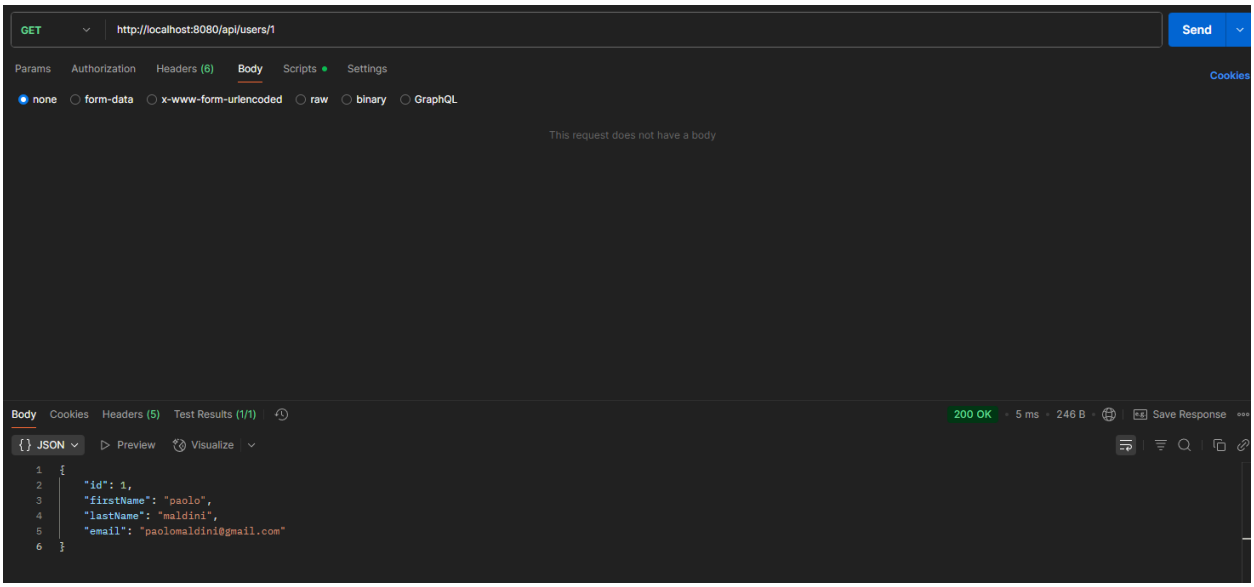
```
spring.datasource.url=jdbc:mysql://localhost:3306/user_management
spring.datasource.username=root
spring.datasource.password=

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
spring.jpa.hibernate.ddl-auto=update

server.port=8080
```

OUTPUT:

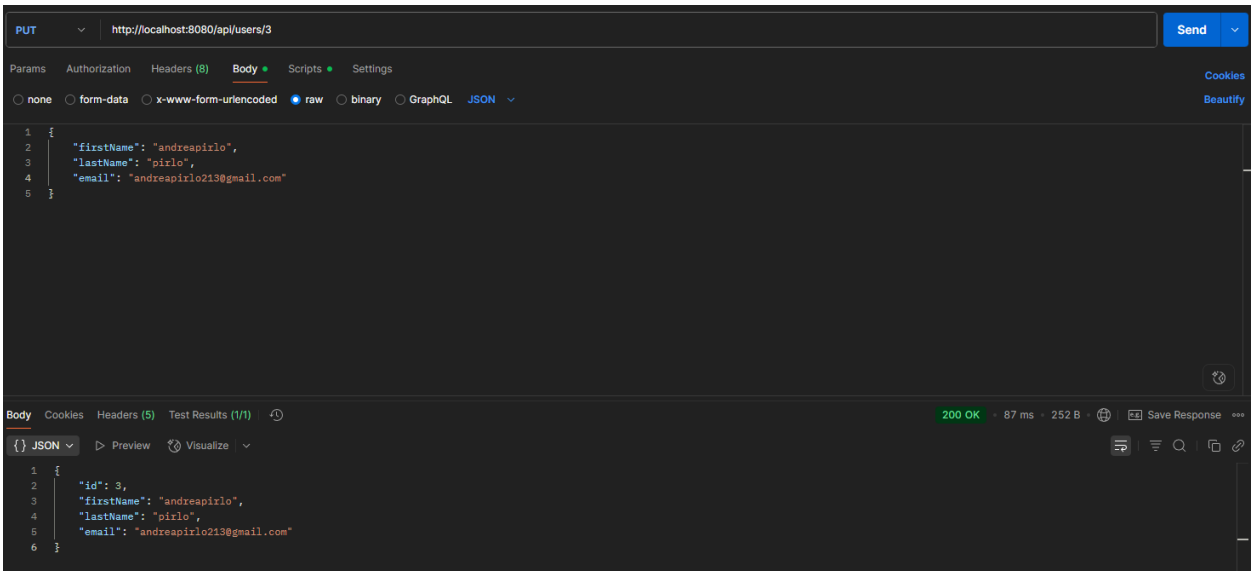
1. **Create User:**



2. Read Single User:



3. Update User:



4. Delete User