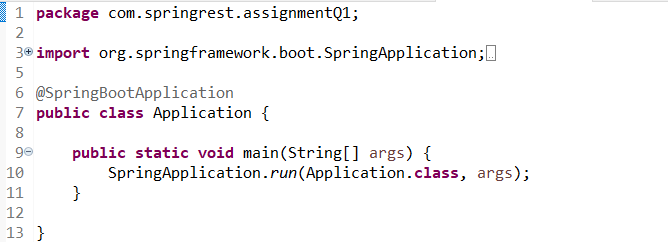
Spring REST Assignments

Submitted by -Sanket Bolamwar

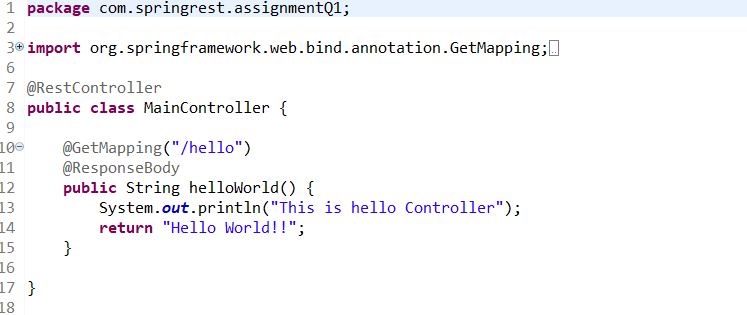
1. Create a RESTful web service that returns "Hello World" message.

**Refere = Spring.REST.AssignmentQ1**

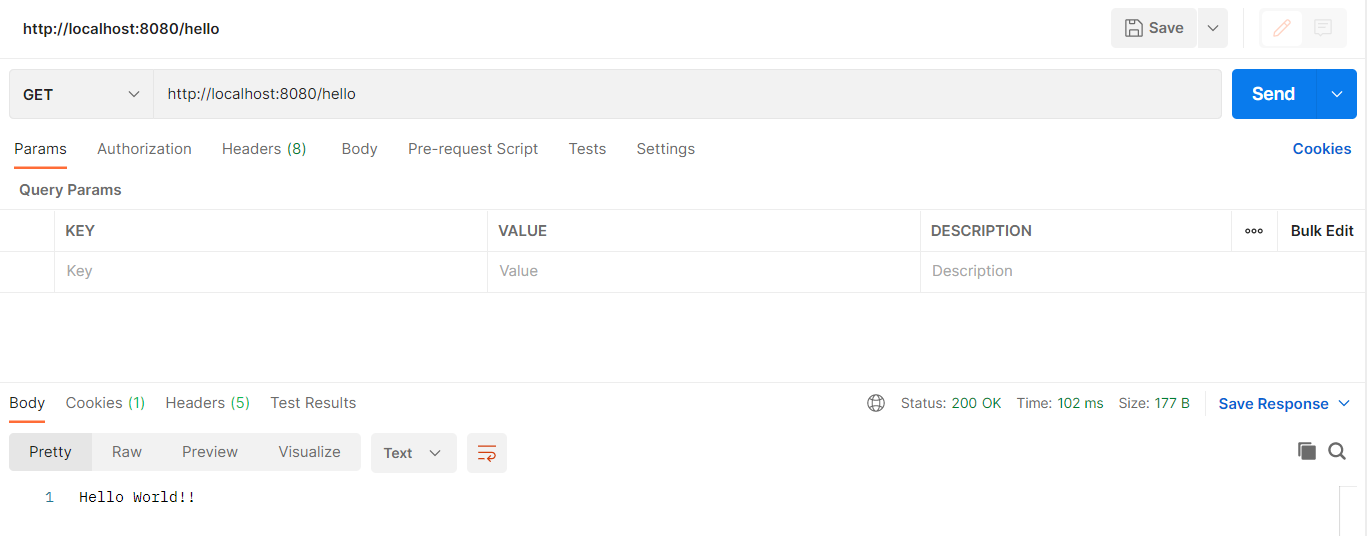
**Application Class:**

****

**MainController Class:**

****

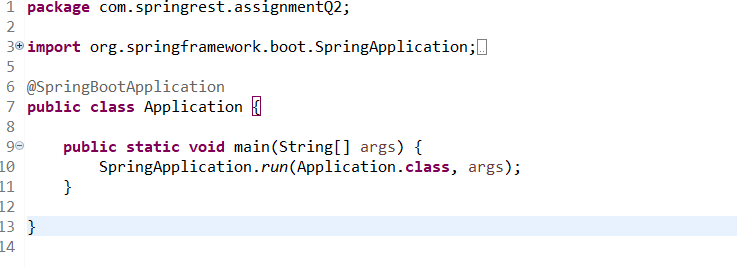
**Output:**

****

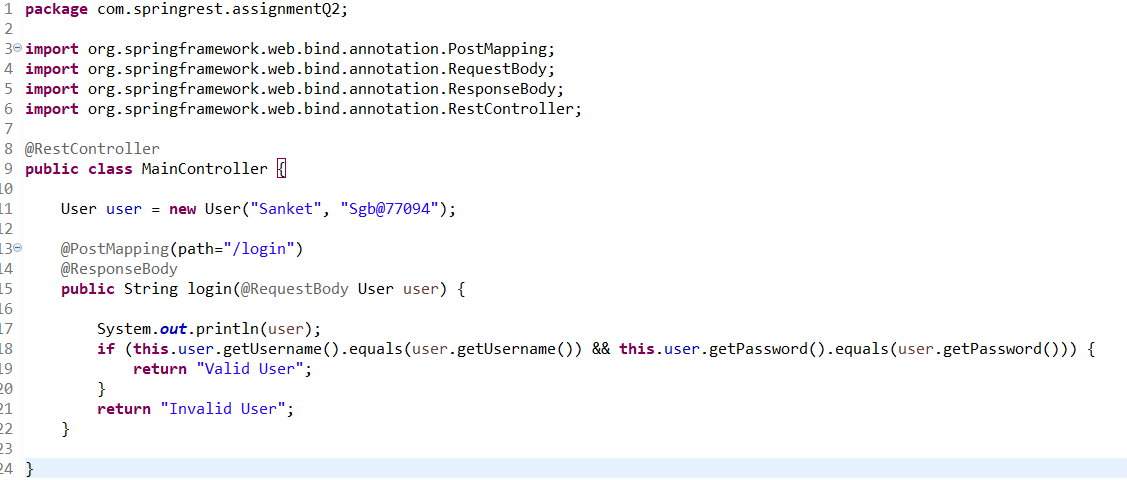
1. Create a RESTful web service that authenticates an user. User will specify his/her credentials i.e. username and password. If username and password are correct, It should return "valid user" message, else "Invalid user" message.

**Refere = Spring.REST.AssignmentQ2**

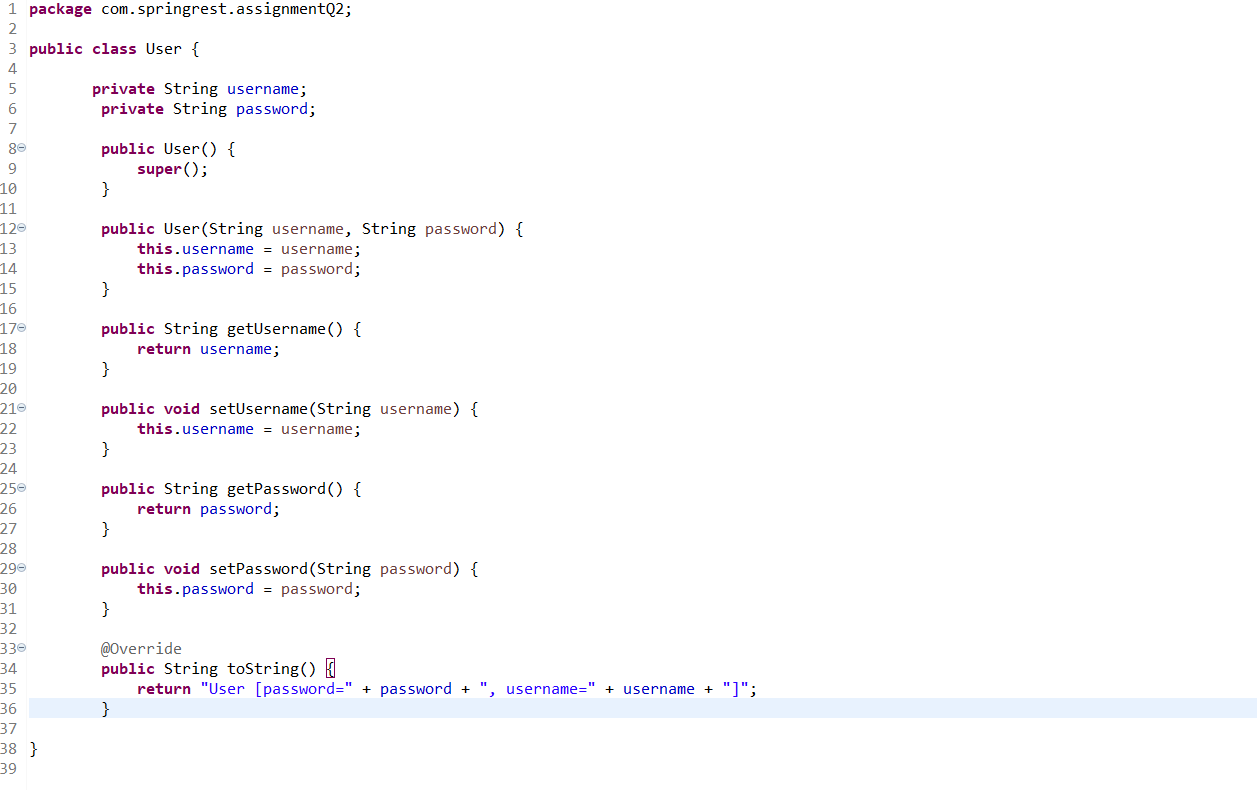
**Application Class:**

****

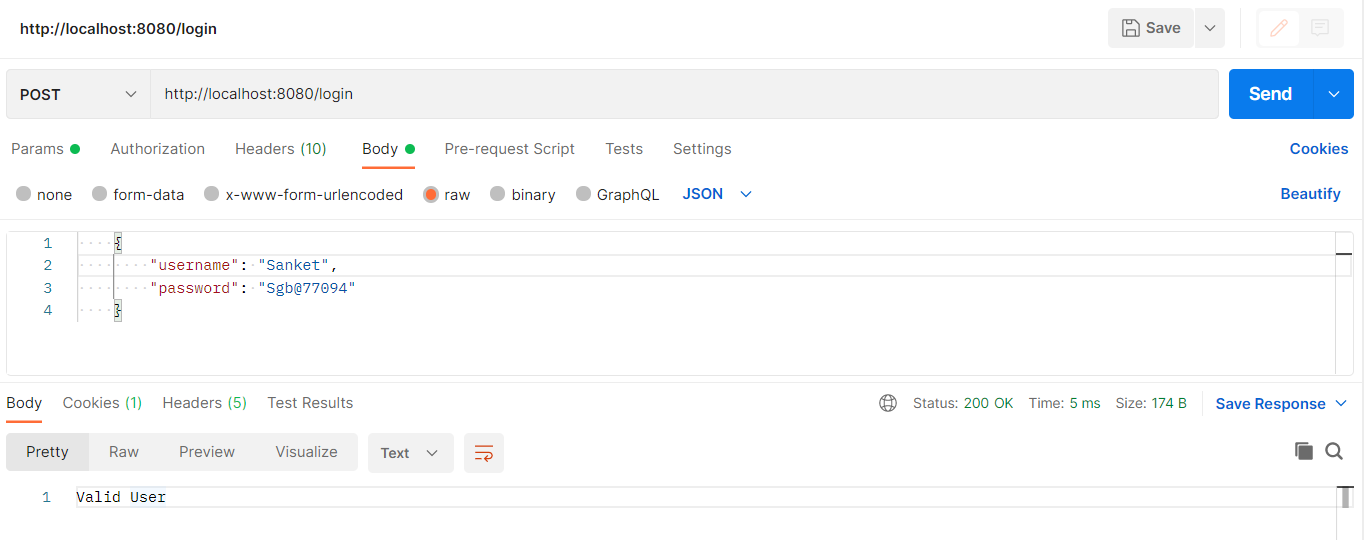
**MainController Class:**

****

**User Class:**

****

**Output:**



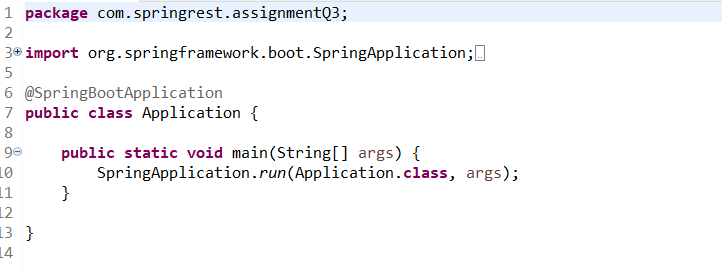
1. Create a RESTful web service that returns state, city and country information when user passes zipcode. You can send state, city and country information in JSON format.

Sample Input: 99501

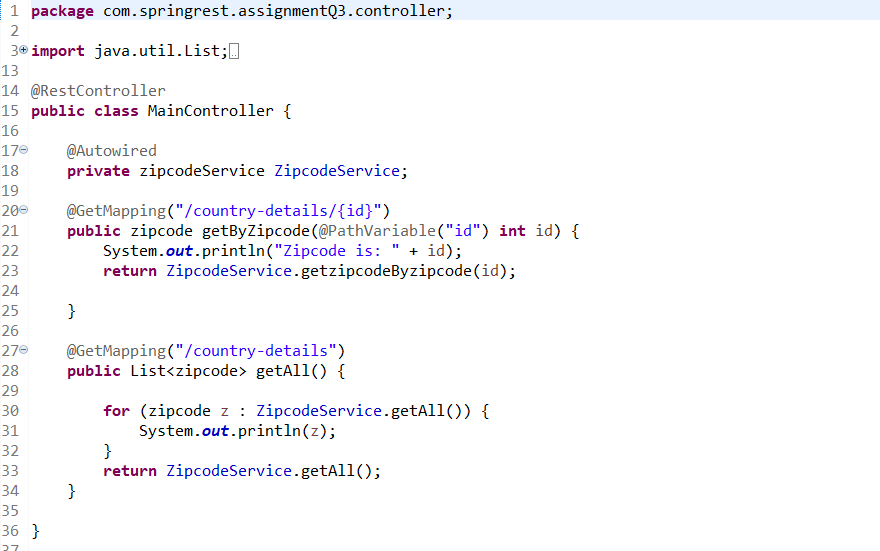
Sample output: {"state": "AK", City: "ANCHORAGE", "country: "US"}

**Refere = Spring.REST.AssignmentQ3**

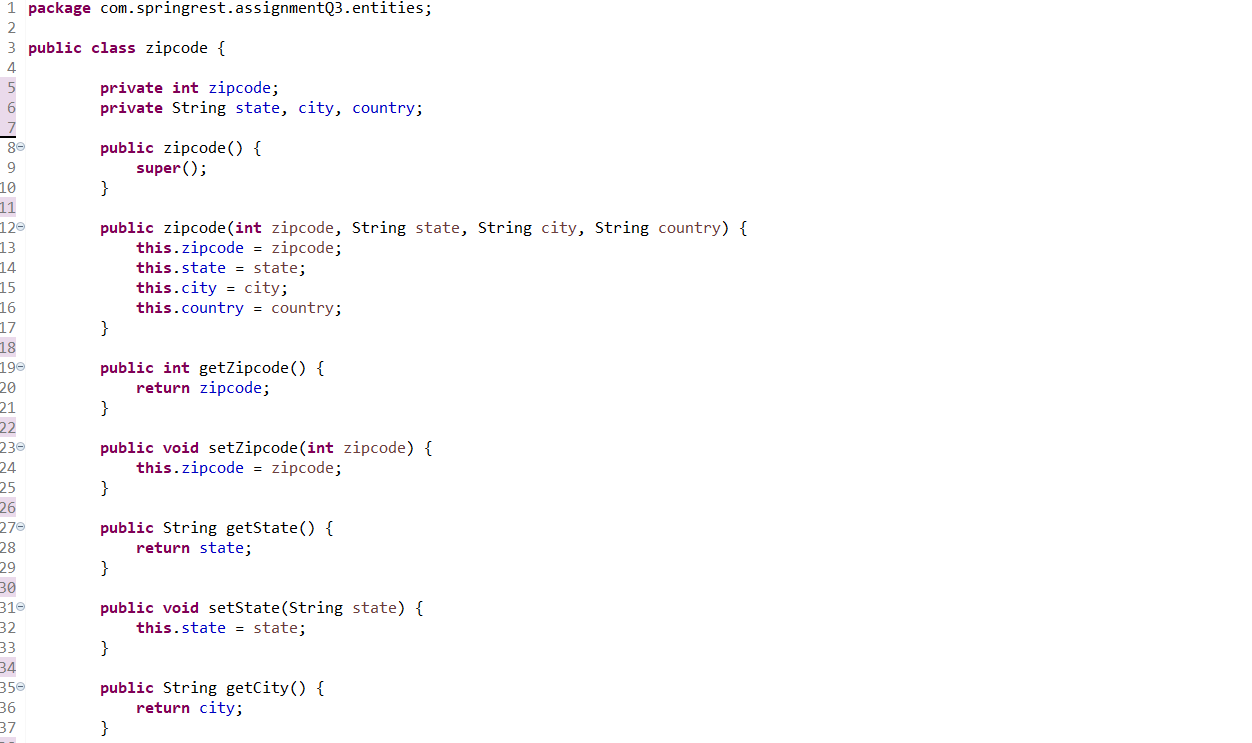
**Application Class:**

****

**MainController Class:**

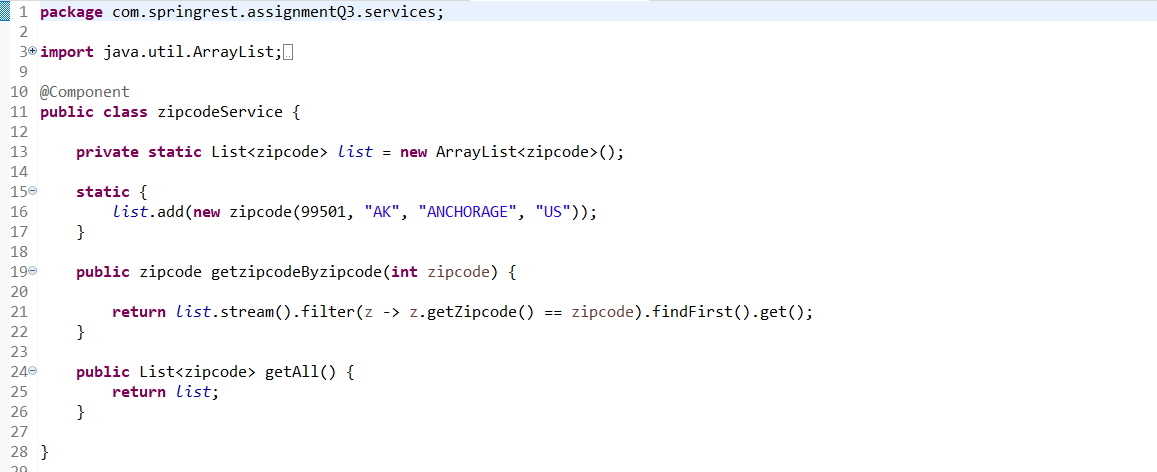
****

**Zipcode Class:**

****

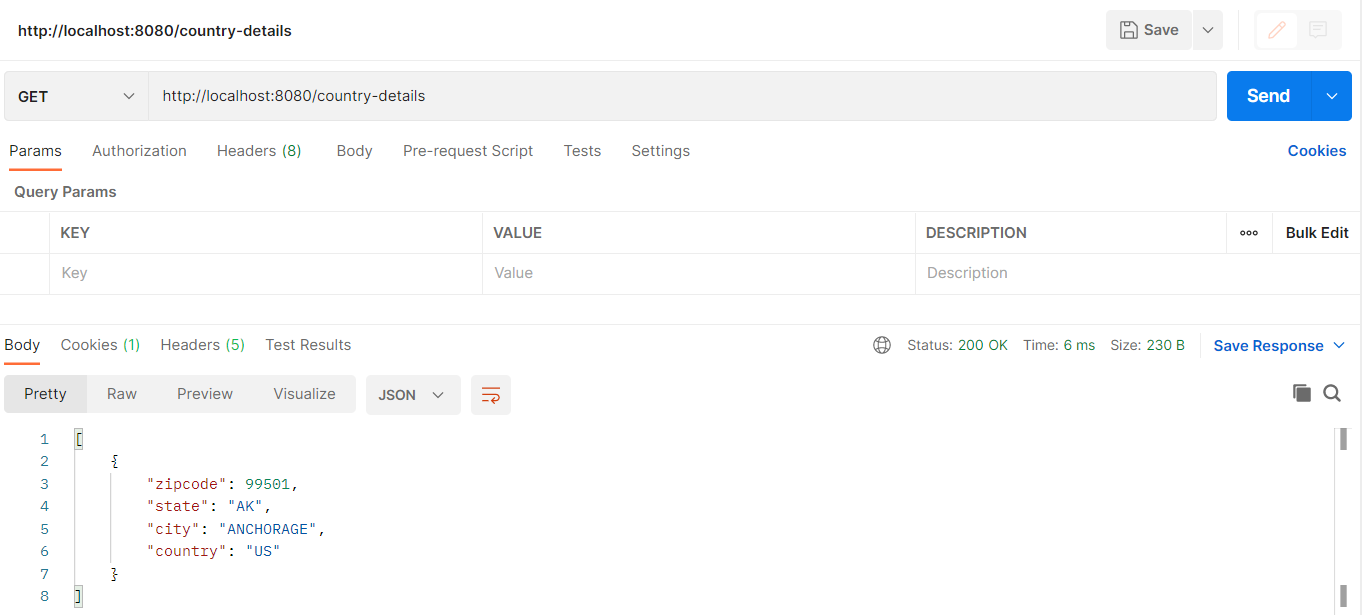
****

**ZipcodeService Class:**

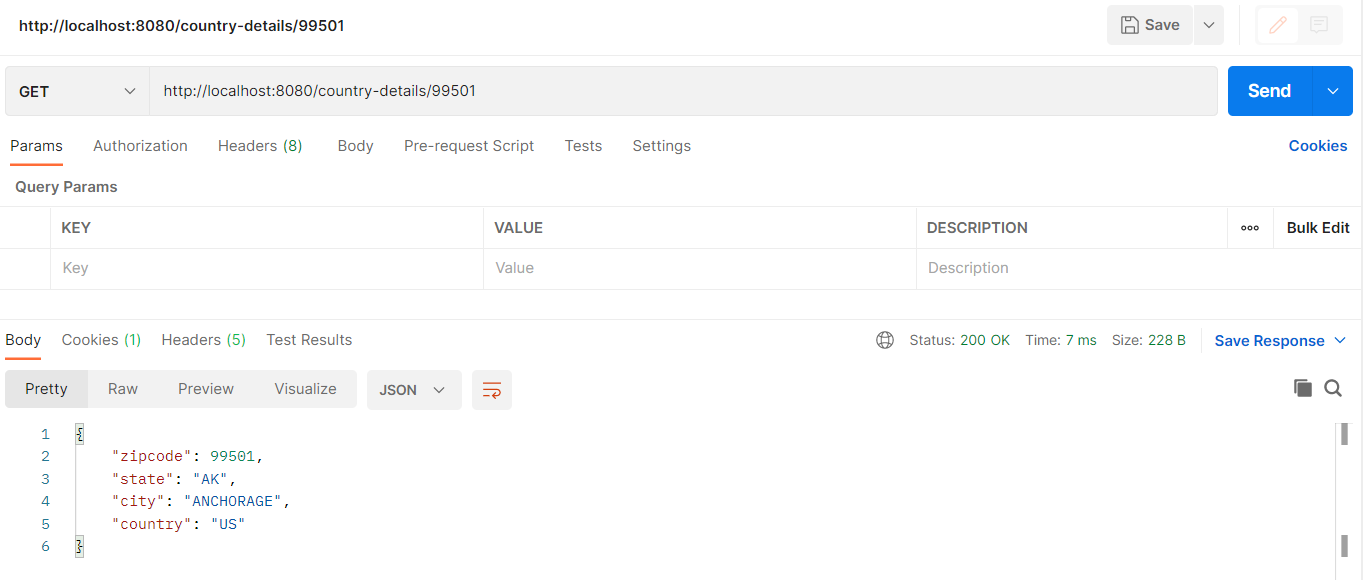


**Output:**

**Using http://localhost:8080/country-details**



**Using http://localhost:8080/country-details/99501**



1. Create a RESTful web service that validates the credit card. It means we need to check the type of credit card like American Express, Discover, Visa etc and it is valid or not.

Valid length: 16 digits.

First 6 digits must be in one of the following ranges:

601100 through 601109

601120 through 601149

601174

601177 through 601179

601186 through 601199

644000 through 659999

enRoute

Valid length: 15 digits. First four digits must be 2014 or 2149.

JCB

Valid length: 16 to 19 digits.

First 4 digits must be in the range 3528 through 3589.

MasterCard

Valid length: 16 digits.

First digit must be 5 and second digit must be in the range 1 through 5 inclusive. The range is 510000 through 559999.

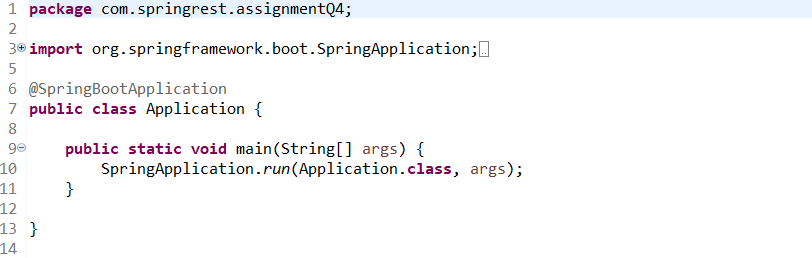
Visa

Valid length: Up to 19 digits. First digit must be a 4.

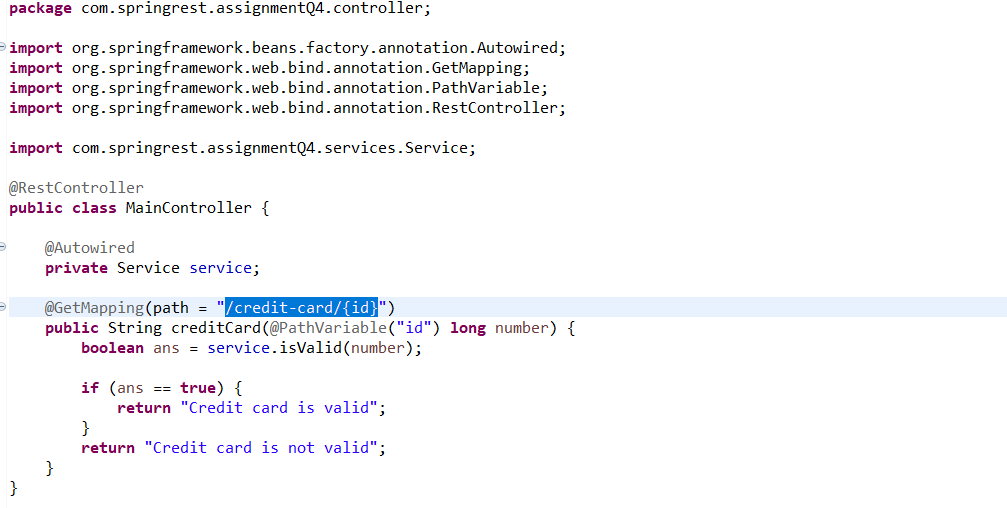
First digit must be 2 and second digit must be in the range 2 through 7 inclusive. The range is 222100 through 272099.

**Refere = Spring.REST.AssignmentQ4**

**Application Class:**

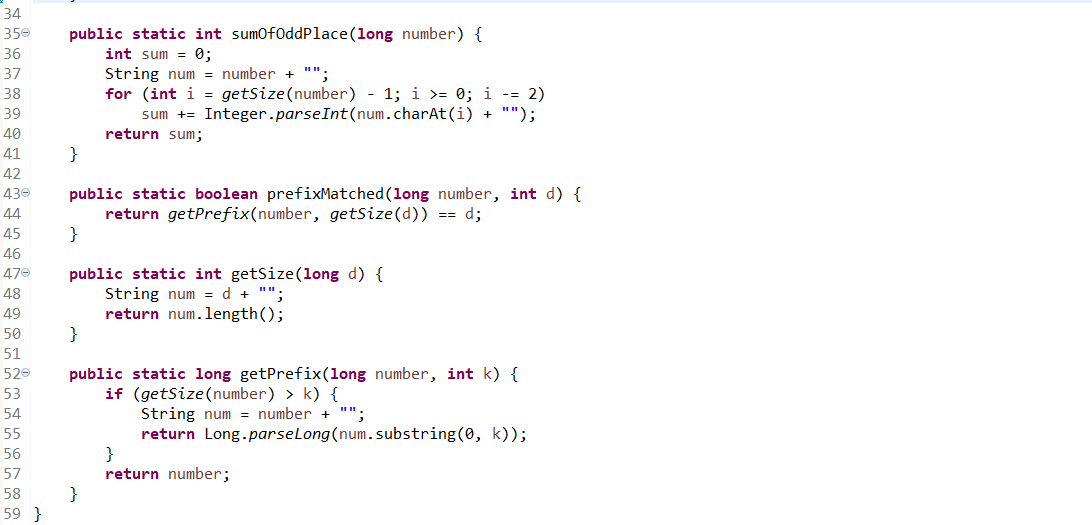
****

**MainController Class:**

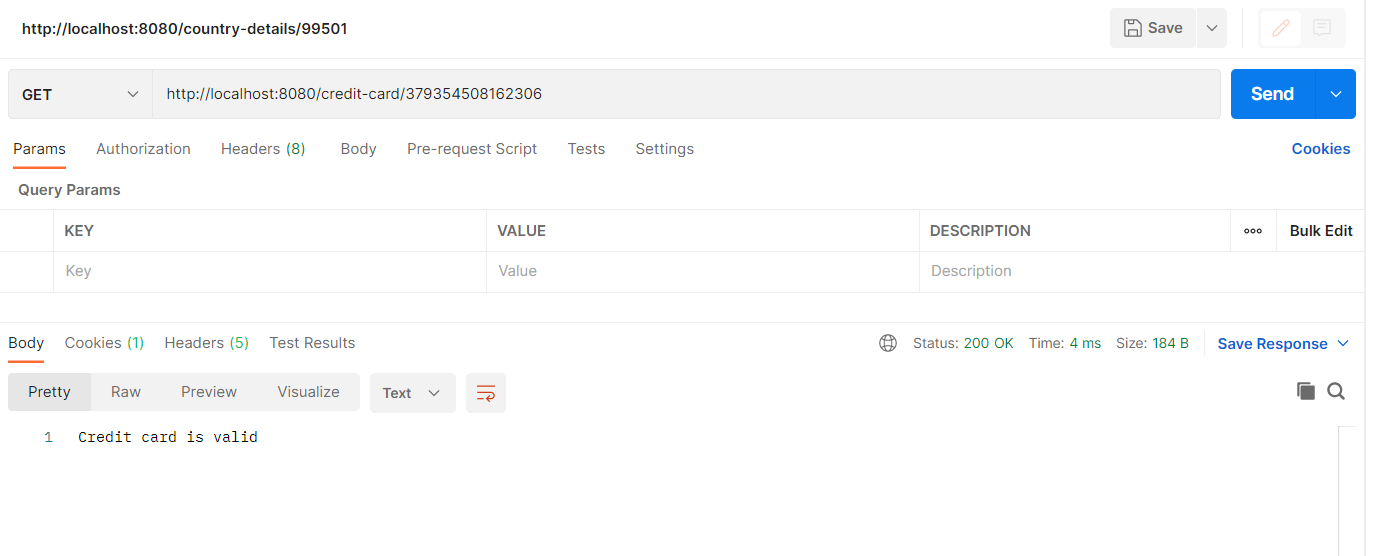
****

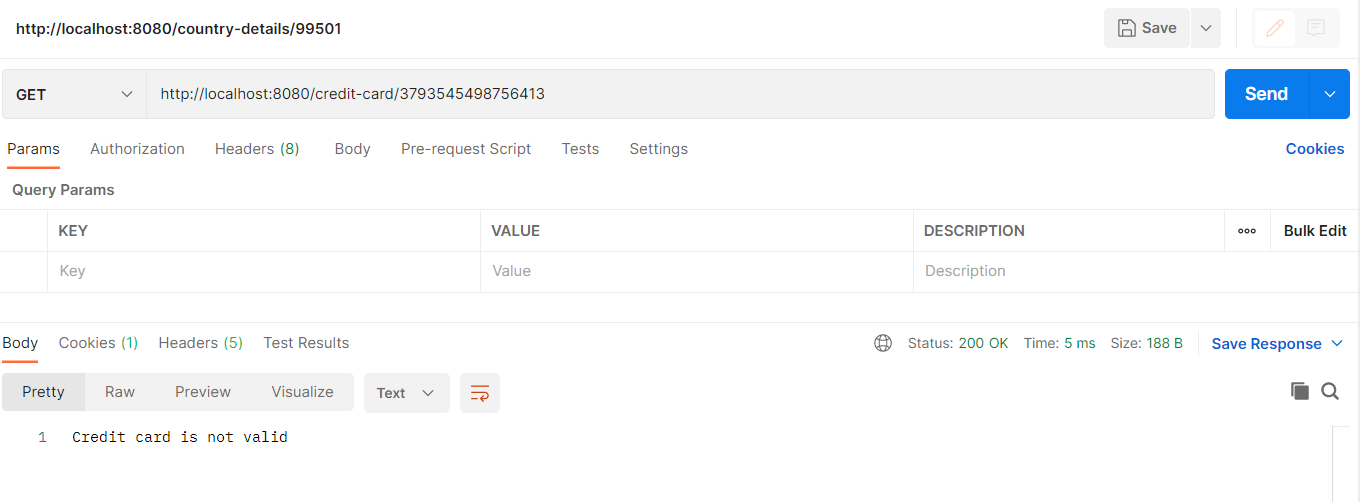
**Service Class:**

****

****

**Output:**

****



1. Develop RESTful web services for "Employee Management System" that manages the information about employees
   1. Add a new employee
   2. Searching for specific employee
   3. Deleting an existing employee
   4. Finding all employees
   5. Editing/updating employee information.

Create a Employee domain model class having following properties: employeeId, employeeName, employeeDepartment, employeeDesignation, employeeSalary. Employee Id should be generated automatically at database level.

Develop controller, service and repository layers classes. Use CrudRepository from Spring Data.

**Refere = Spring.REST.AssignmentQ5**

**Application Class:**

package com.springrest.assignmentQ5.services;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

import com.springrest.assignmentQ5.Repository.EmployeeRepo;

import com.springrest.assignmentQ5.entities.Employee;

@Component

public class EmployeeService {

@Autowired

private EmployeeRepo employeeRepo;

public void setEmployee(Employee employee) {

employeeRepo.save(employee);

System.out.println("Employee added successfully");

}

public void setAllEmployee(List<Employee> employees) {

employeeRepo.saveAll(employees);

System.out.println("Added all the employees");

}

public List<Employee> findAllEmployees() {

return employeeRepo.findAll();

}

public Employee findEmployeeById(int id) {

return employeeRepo.findById(id);

}

public void updateEmployee(Employee e, int id) {

Employee employee = employeeRepo.findById(id);

employee.setName(e.getName());

employee.setDepartment(e.getDepartment());

employee.setDesignation(e.getDesignation());

employee.setSalary(e.getSalary());

employeeRepo.save(employee);

System.out.println("Employee Updated Successfully");

}

public void deleteEmployee(int id) {

employeeRepo.deleteById(id);

System.out.println("Employee Deleted Successfully");

}

}

**EmployeeController Class:**

package com.springrest.assignmentQ5.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RestController;

import com.springrest.assignmentQ5.entities.Employee;

import com.springrest.assignmentQ5.services.EmployeeService;

@RestController

public class EmployeeController {

@Autowired

private EmployeeService employeeService;

@PostMapping("/employee")

public String AddEmployee(@RequestBody Employee employee) {

System.out.println(employee);

employeeService.setEmployee(employee);

return "Employee Added Successfully";

}

@GetMapping("/employee")

public List<Employee> getAllEmployees() {

return employeeService.findAllEmployees();

}

@GetMapping("/employee/{id}")

public Employee getEmployee(@PathVariable("id") int id) {

return employeeService.findEmployeeById(id);

}

@DeleteMapping("/employee/{id}")

public String deleteEmployeeString(@PathVariable("id") int id) {

employeeService.deleteEmployee(id);

return "Employee Deleted Successfully";

}

@PutMapping("/employee/{id}")

public String updateEmployee(@RequestBody Employee e, @PathVariable("id") int id) {

employeeService.updateEmployee(e, id);

return "Employee Updated Successfully";

}

}

**Employee Class:**

package com.springrest.assignmentQ5.entities;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table(name = "employee")

public class Employee {

@Id

private int id;

private String name;

private String department;

private String designation;

private long salary;

public Employee() {

super();

}

public Employee(int id, String name, String department, String designation, long salary) {

this.id = id;

this.name = name;

this.department = department;

this.designation = designation;

this.salary = salary;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getDepartment() {

return department;

}

public void setDepartment(String department) {

this.department = department;

}

public String getDesignation() {

return designation;

}

public void setDesignation(String designation) {

this.designation = designation;

}

public long getSalary() {

return salary;

}

public void setSalary(long salary) {

this.salary = salary;

}

@Override

public String toString() {

return "Employee [department=" + department + ", designation=" + designation + ", id=" + id + ", name=" + name

+ ", salary=" + salary + "]";

}

}

**EmployeeRepo Class:**

package com.springrest.assignmentQ5.Repository;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import com.springrest.assignmentQ5.entities.Employee;

@Repository

public interface EmployeeRepo extends JpaRepository<Employee, Integer> {

public Employee findById(int id);

}

**EmployeeService Class:**

package com.springrest.assignmentQ5.services;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

import com.springrest.assignmentQ5.Repository.EmployeeRepo;

import com.springrest.assignmentQ5.entities.Employee;

@Component

public class EmployeeService {

@Autowired

private EmployeeRepo employeeRepo;

public void setEmployee(Employee employee) {

employeeRepo.save(employee);

System.out.println("Employee added successfully");

}

public void setAllEmployee(List<Employee> employees) {

employeeRepo.saveAll(employees);

System.out.println("Added all the employees");

}

public List<Employee> findAllEmployees() {

return employeeRepo.findAll();

}

public Employee findEmployeeById(int id) {

return employeeRepo.findById(id);

}

public void updateEmployee(Employee e, int id) {

Employee employee = employeeRepo.findById(id);

employee.setName(e.getName());

employee.setDepartment(e.getDepartment());

employee.setDesignation(e.getDesignation());

employee.setSalary(e.getSalary());

employeeRepo.save(employee);

System.out.println("Employee Updated Successfully");

}

public void deleteEmployee(int id) {

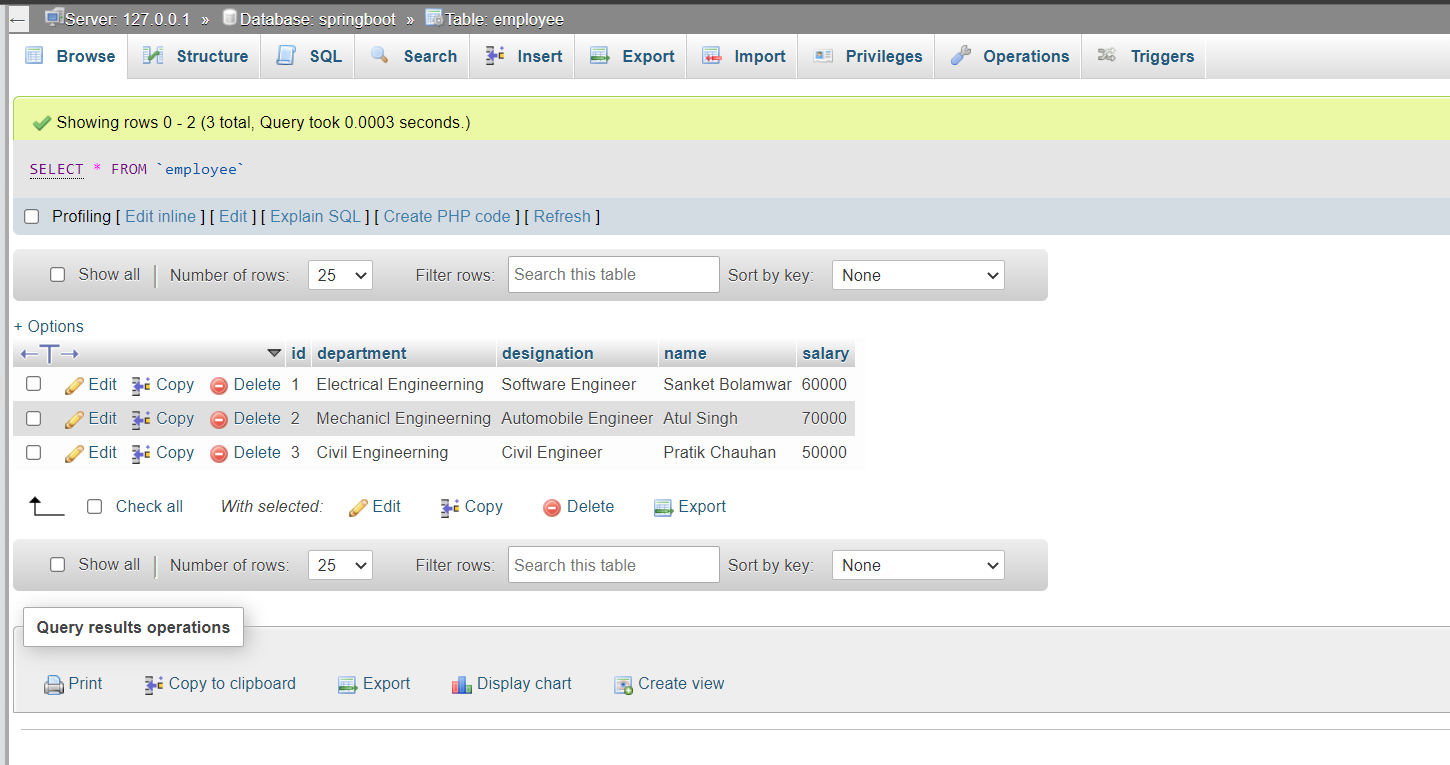
employeeRepo.deleteById(id);

System.out.println("Employee Deleted Successfully");

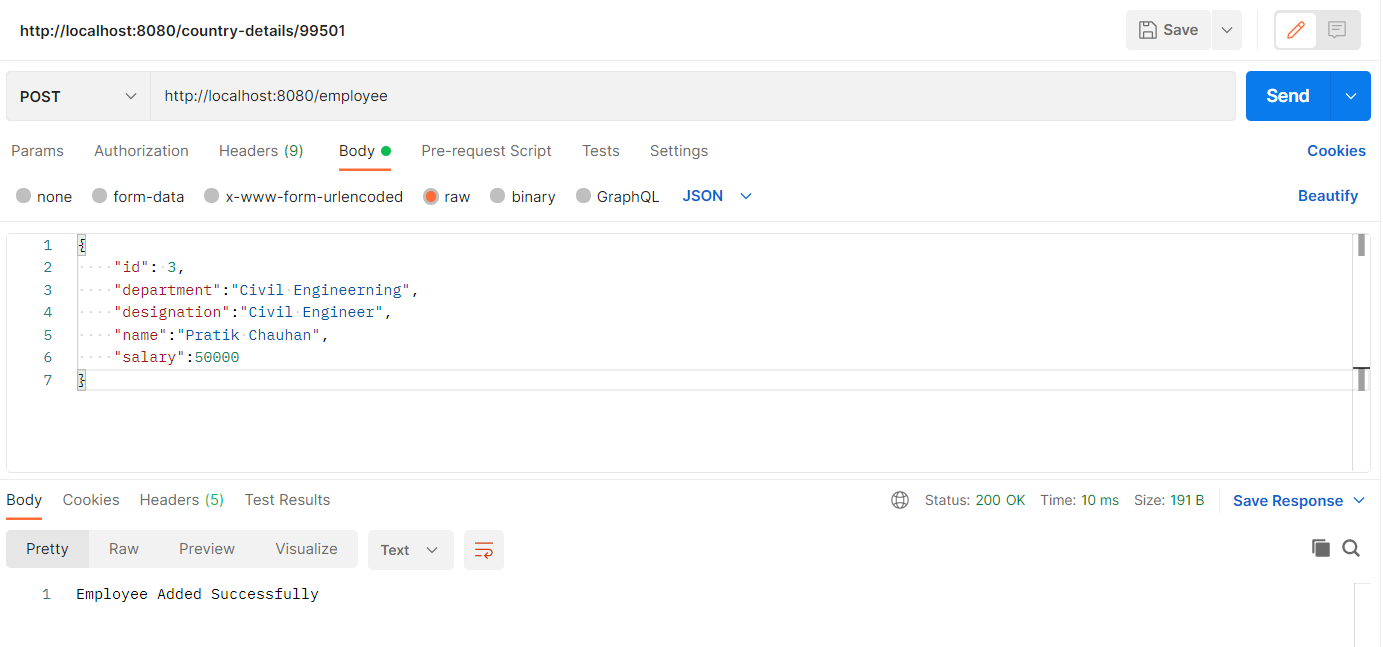
}

}

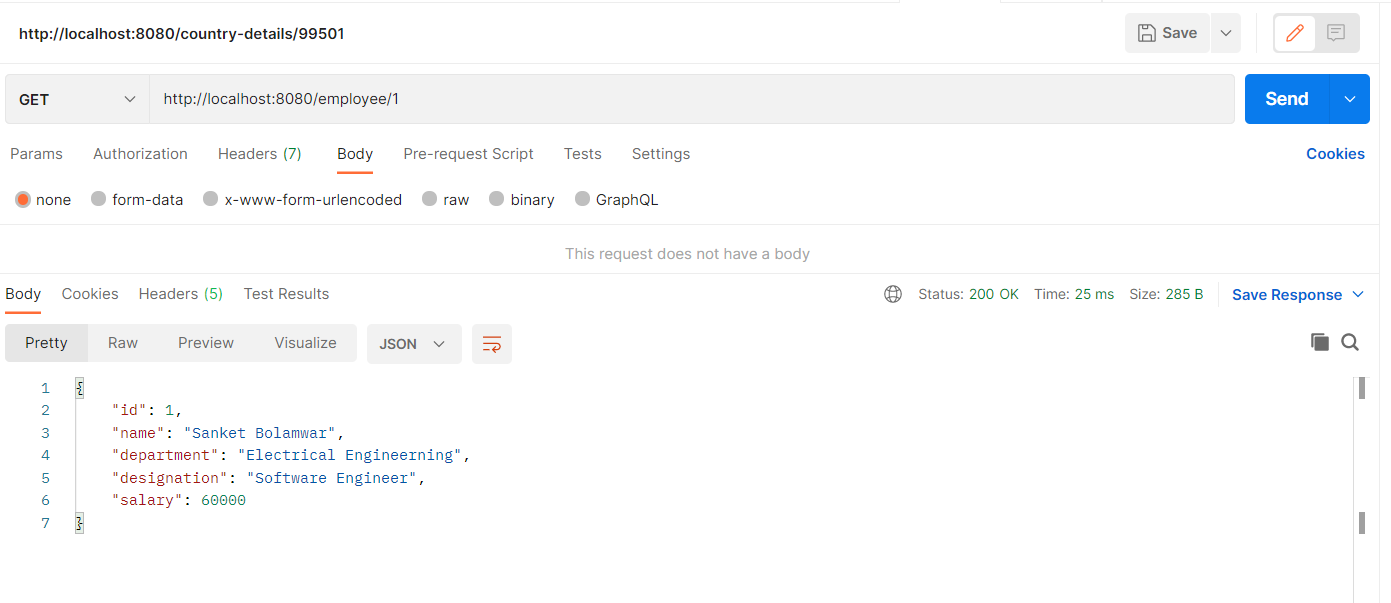
**Database:**

****

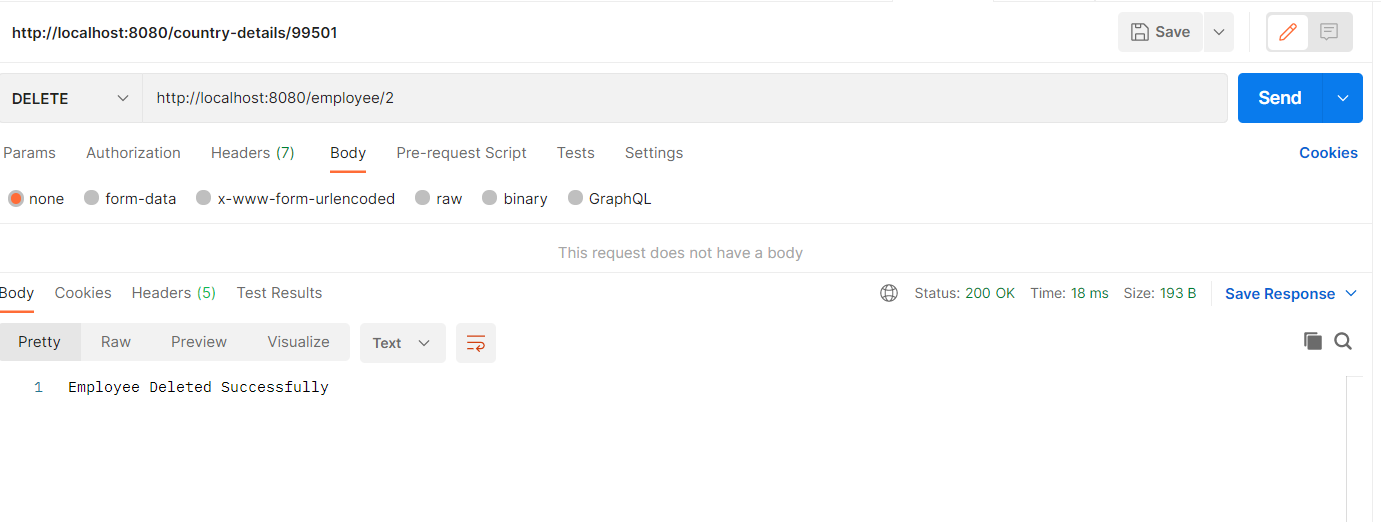
1. **Add a new Employee:**

****

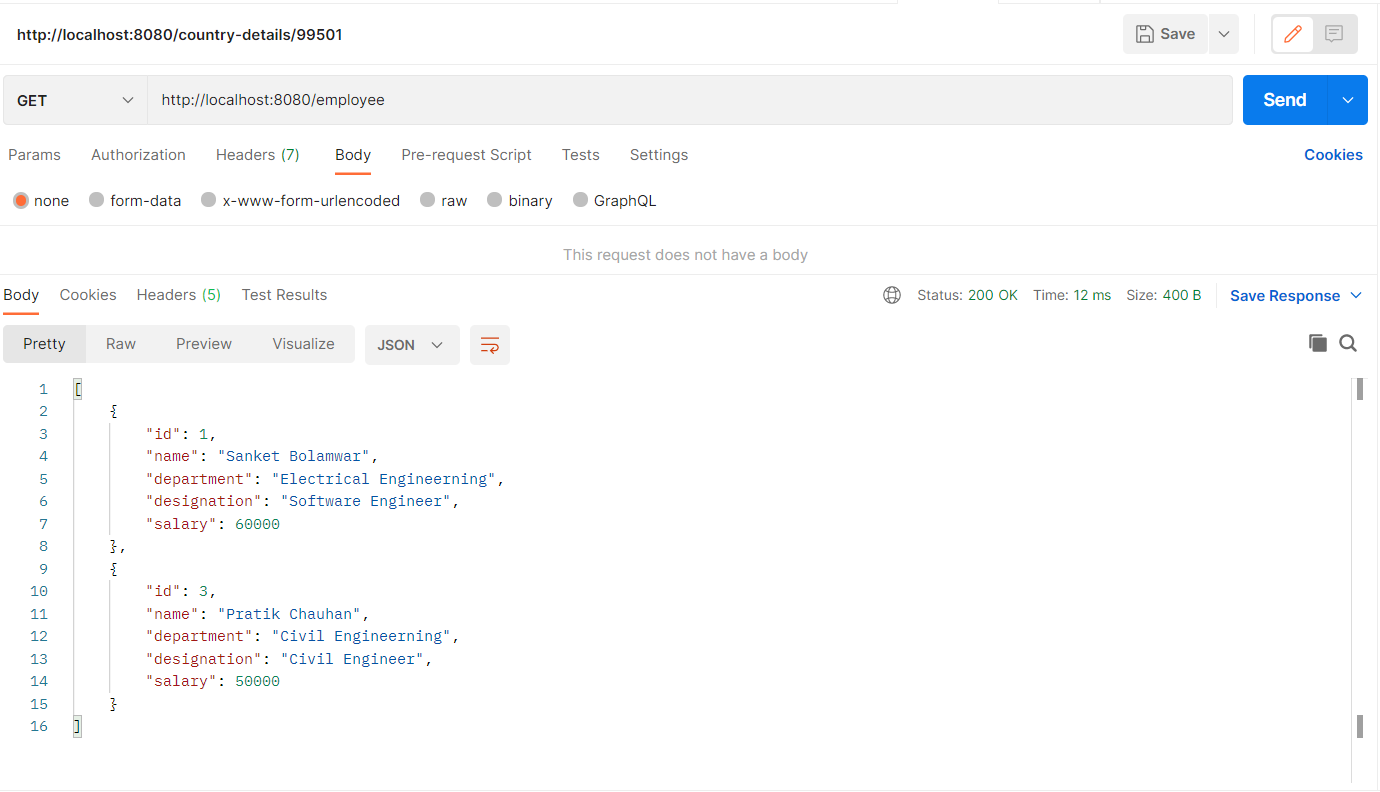
1. **Search for a specific Employee:**

****

1. **Deleting an existing employee**

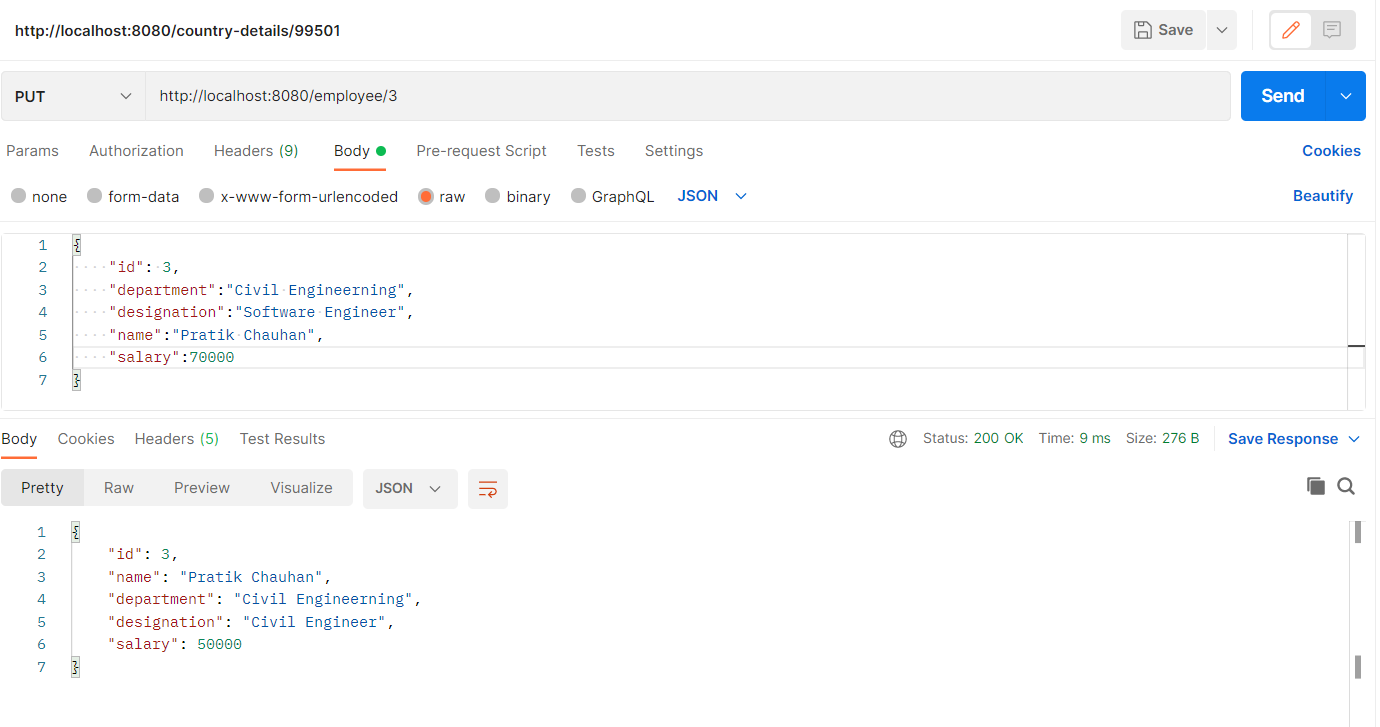
****

1. **Finding all employees**

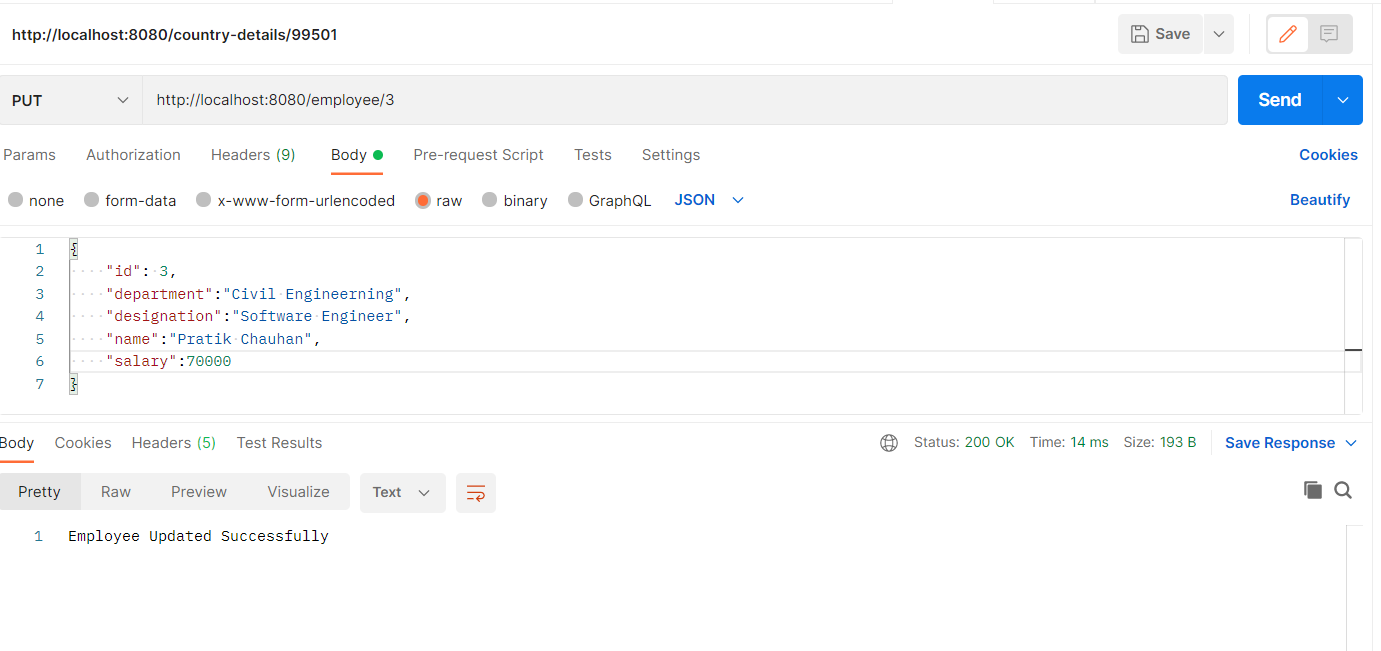
****

1. **Editing/updating employee information.**

**Before Updating**

****

**After Updating**

****

1. Create a Calculator RESTful service that provides following functionality.
   1. Addition of the 2 numbers
   2. Subtraction of the 2 numbers
   3. Multiplication of 2 numbers
   4. Division of 2 numbers
   5. Finding square root of a number.

Consume the above RESTful web service by using RestTemplate.

**Refere = Spring.REST.AssignmentQ6**

**Application Class:**

package com.springrest.assignmentQ6;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class Application {

public static void main(String[] args) {

SpringApplication.run(Application.class, args);

}

}

**MainController Class:**

**package** com.springrest.assignmentQ6.controller;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.web.bind.annotation.GetMapping;

**import** org.springframework.web.bind.annotation.PathVariable;

**import** org.springframework.web.bind.annotation.ResponseBody;

**import** com.springrest.assignmentQ6.service.service;

@Controller

@ResponseBody

**public** **class** MainController {

// 1. Addition of the 2 numbers

// 2. Subtraction of the 2 numbers

// 3. Multiplication of 2 numbers

// 4. Division of 2 numbers

// 5. Finding square root of a number.

@Autowired

**private** service s;

@GetMapping("/addition/{num1}/{num2}")

**public** String Addition(@PathVariable("num1") **int** num1, @PathVariable("num2") **int** num2) {

System.***out***.println("Number 1 = " + num1 + "/nNumber 2 = " + num2);

**return** ("Addition of " + num1 + " and " + num2 + " = " + s.Addition(num1, num2));

}

@GetMapping("/subtraction/{num1}/{num2}")

**public** String Subtraction(@PathVariable("num1") **int** num1, @PathVariable("num2") **int** num2) {

**return** ("Subtraction of " + num1 + " by " + num2 + " = " + s.subtraction(num1, num2));

}

@GetMapping("/multiplication/{num1}/{num2}")

**public** String Multiplication(@PathVariable("num1") **int** num1, @PathVariable("num2") **int** num2) {

**return** ("Multiplication of " + num1 + " and " + num2 + " = " + s.multiplication(num1, num2));

}

@GetMapping("/division/{num1}/{num2}")

**public** String Division(@PathVariable("num1") **int** num1, @PathVariable("num2") **int** num2) {

**return** ("Division of " + num1 + " and " + num2 + " = " + s.division(num1, num2));

}

@GetMapping("/square-root/{num1}")

**public** String SquareRoot(@PathVariable("num1") **int** num1) {

**return** ("Addition of " + num1 + " = " + s.square(num1));

}

}

**Service Class:**

**package** com.springrest.assignmentQ6.service;

**import** org.springframework.stereotype.Component;

@Component

**public** **class** service {

**public** **int** Addition(**int** a, **int** b) {

System.***out***.println(a + b);

**return** a + b;

}

**public** **int** subtraction(**int** a, **int** b) {

System.***out***.println(a - b);

**return** a - b;

}

**public** **int** multiplication(**int** a, **int** b) {

System.***out***.println(a \* b);

**return** a \* b;

}

**public** **double** division(**int** a, **int** b) {

**try** {

System.***out***.println(a / b);

**return** a / b;

} **catch** (Exception e) {

e.printStackTrace();

**return** -1;

}

}

**public** **double** square(**int** a) {

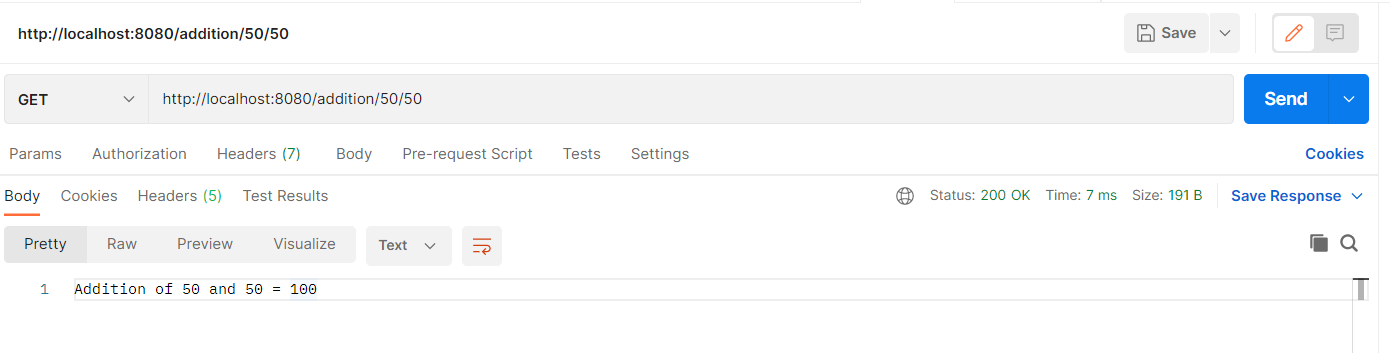
System.***out***.println(Math.*sqrt*(a));

**return** Math.*sqrt*(a);

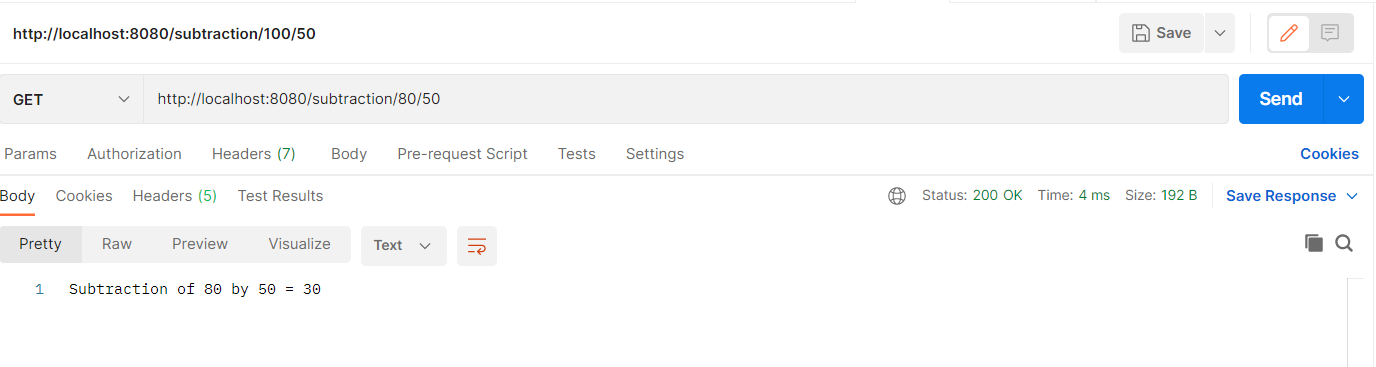
}

}

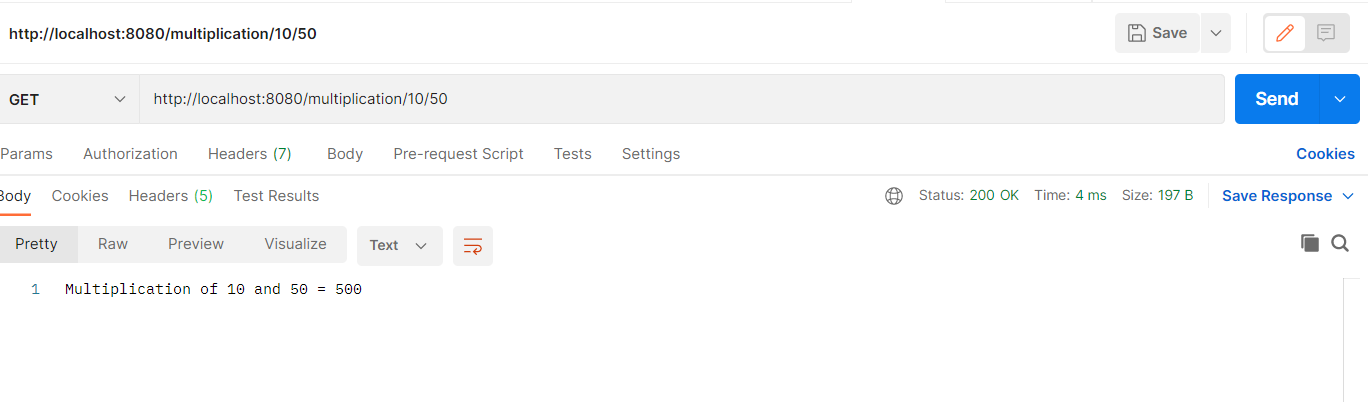
* 1. **Addition of the 2 numbers**

****

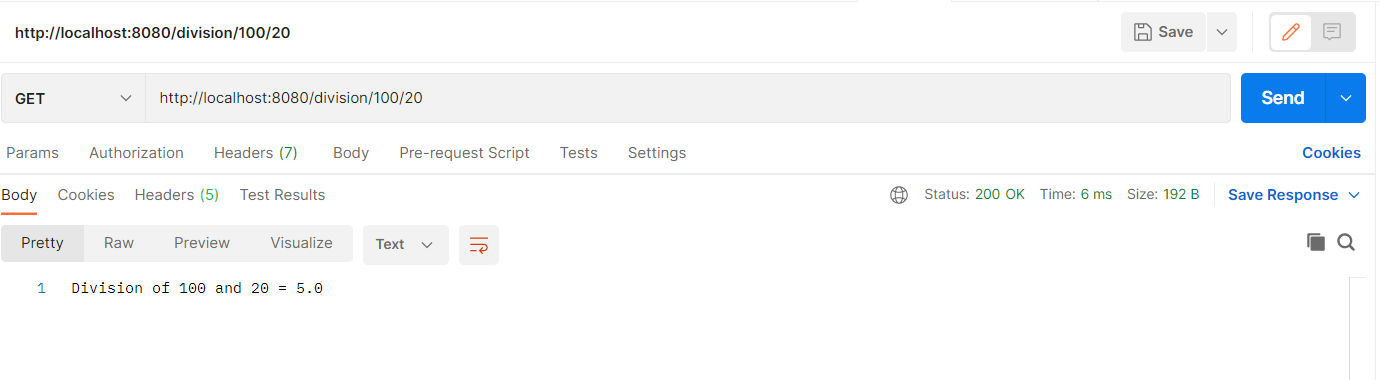
* 1. **Subtraction of the 2 numbers**

****

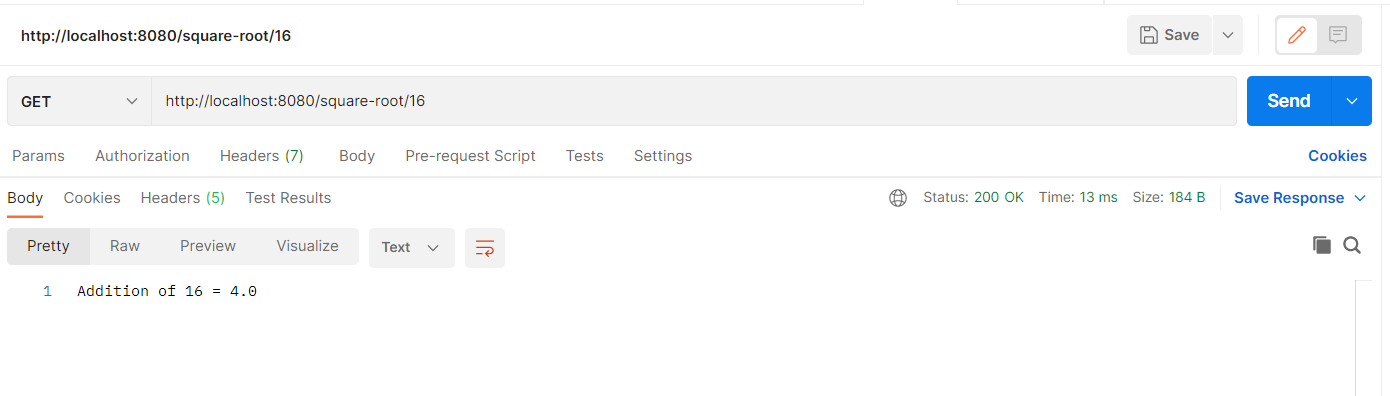
* 1. **Multiplication of 2 numbers**

****

* 1. **Division of 2 numbers**

****

* 1. **Finding square root of a number.**



1. Design and develop RESTful web service as follows:
   1. A user can place an order
   2. A user can update an order
   3. A user can view specific order
   4. A user can view all the orders
   5. A user can delete a specific order.

Note: Use MongoRepository of Spring data to store order details

**Refere = Spring.REST.AssignmentQ7**

**Application Class:**

package com.springrest.assignmentQ7;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class Application {

public static void main(String[] args) {

SpringApplication.run(Application.class, args);

}

}

**OrderController Class:**

**package** com.springrest.assignmentQ7.controller;

**import** java.util.List;

**import** java.util.Optional;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.http.HttpStatus;

**import** org.springframework.http.ResponseEntity;

**import** org.springframework.web.bind.annotation.DeleteMapping;

**import** org.springframework.web.bind.annotation.GetMapping;

**import** org.springframework.web.bind.annotation.PathVariable;

**import** org.springframework.web.bind.annotation.PostMapping;

**import** org.springframework.web.bind.annotation.PutMapping;

**import** org.springframework.web.bind.annotation.RequestBody;

**import** org.springframework.web.bind.annotation.RequestParam;

**import** org.springframework.web.bind.annotation.RestController;

**import** com.springrest.assignmentQ7.entities.Order;

**import** com.springrest.assignmentQ7.services.OrderService;

@RestController

**public** **class** OrderController {

@Autowired

**private** OrderService orderService;

// To add orders

@PostMapping("/orders")

**public** ResponseEntity<Order> addOrder(@RequestBody Order order) {

**try** {

System.***out***.println(order);

orderService.addOrder(order);

**return** ResponseEntity.*ok*(order);

} **catch** (Exception e) {

e.printStackTrace();

**return** ResponseEntity.*status*(HttpStatus.***INTERNAL\_SERVER\_ERROR***).build();

}

}

// To read all the orders

@GetMapping("/orders")

**public** ResponseEntity<List<Order>> getAllOrder() {

List<Order> list = orderService.getAllOrders();

**if** (list.size() <= 0) {

**return** ResponseEntity.*status*(HttpStatus.***NOT\_FOUND***).build();

}

**return** ResponseEntity.*of*(Optional.*of*(list));

}

// Find all the orders by id.

@GetMapping("/orders/{id}")

**public** ResponseEntity<Order> getOrderById(@PathVariable("id") **int** id) {

**try** {

Order order = orderService.getOrderById(id);

**return** ResponseEntity.*of*(Optional.*of*(order));

} **catch** (Exception e) {

e.printStackTrace();

**return** ResponseEntity.*status*(HttpStatus.***NOT\_FOUND***).build();

}

}

// Find all the orders by name.

@GetMapping("/orders/name")

**public** ResponseEntity<Order> getOrderByName(@RequestParam(value = "name") String name) {

**try** {

Order order = orderService.getOrderByName(name);

**return** ResponseEntity.*of*(Optional.*of*(order));

} **catch** (Exception e) {

e.printStackTrace();

**return** ResponseEntity.*status*(HttpStatus.***NOT\_FOUND***).build();

}

}

// To update the order

@PutMapping("/orders/{id}")

**public** ResponseEntity<Order> updateOrder(@RequestBody Order order, @PathVariable("id") **int** id) {

Order order1 = **null**;

**try** {

order1 = orderService.updateOrder(order);

**return** ResponseEntity.*ok*().body(order1);

} **catch** (Exception e) {

e.printStackTrace();

**return** ResponseEntity.*status*(HttpStatus.***INTERNAL\_SERVER\_ERROR***).build();

}

}

// To delete the order

@DeleteMapping("/orders/{id}")

**public** ResponseEntity<Order> deleteOrder(@PathVariable("id") **int** id) {

**try** {

Order order = orderService.getOrderById(id);

orderService.deleteOrder(id);

**return** ResponseEntity.*ok*().body(order);

} **catch** (Exception e) {

e.printStackTrace();

**return** ResponseEntity.*status*(HttpStatus.***INTERNAL\_SERVER\_ERROR***).build();

}

}

}

**OrderRepo Class:**

**package** com.springrest.assignmentQ7.dao;

**import** org.springframework.data.mongodb.repository.MongoRepository;

**import** org.springframework.stereotype.Repository;

**import** com.springrest.assignmentQ7.entities.Order;

@Repository

**public** **interface** OrderRepo **extends** MongoRepository<Order, Integer> {

**public** Order findById(**int** id);

**public** Order findByName(String name);

}

**Order Class:**

**package** com.springrest.assignmentQ7.entities;

**import** org.springframework.data.mongodb.core.mapping.Document;

@Document(collection = "orders")

**public** **class** Order {

**private** **int** id;

**private** String name;

**private** String quantity;

**private** String price;

**private** String payment;

**public** Order() {

**super**();

}

**public** Order(**int** id, String name, String quantity, String price, String payment) {

**this**.id = id;

**this**.name = name;

**this**.quantity = quantity;

**this**.price = price;

**this**.payment = payment;

}

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getQuantity() {

**return** quantity;

}

**public** **void** setQuantity(String quantity) {

**this**.quantity = quantity;

}

**public** String getPrice() {

**return** price;

}

**public** **void** setPrice(String price) {

**this**.price = price;

}

**public** String getPayment() {

**return** payment;

}

**public** **void** setPayment(String payment) {

**this**.payment = payment;

}

@Override

**public** String toString() {

**return** "Order [id=" + id + ", name=" + name + ", payment=" + payment + ", price=" + price + ", quantity="

+ quantity + "]";

}

}

**OrderService Class:**

**package** com.springrest.assignmentQ7.services;

**import** java.util.List;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Component;

**import** com.springrest.assignmentQ7.dao.OrderRepo;

**import** com.springrest.assignmentQ7.entities.Order;

@Component

**public** **class** OrderService {

@Autowired

**private** OrderRepo orderRepo;

**public** **void** addOrder(Order order) {

System.***out***.println(order);

orderRepo.save(order);

System.***out***.println("Order Added Successfully");

}

**public** List<Order> getAllOrders() {

**return** orderRepo.findAll();

}

**public** Order getOrderById(**int** id) {

**return** orderRepo.findById(id);

}

**public** Order getOrderByName(String name) {

**return** orderRepo.findByName(name);

}

**public** Order updateOrder(Order order) {

**return** orderRepo.save(order);

}

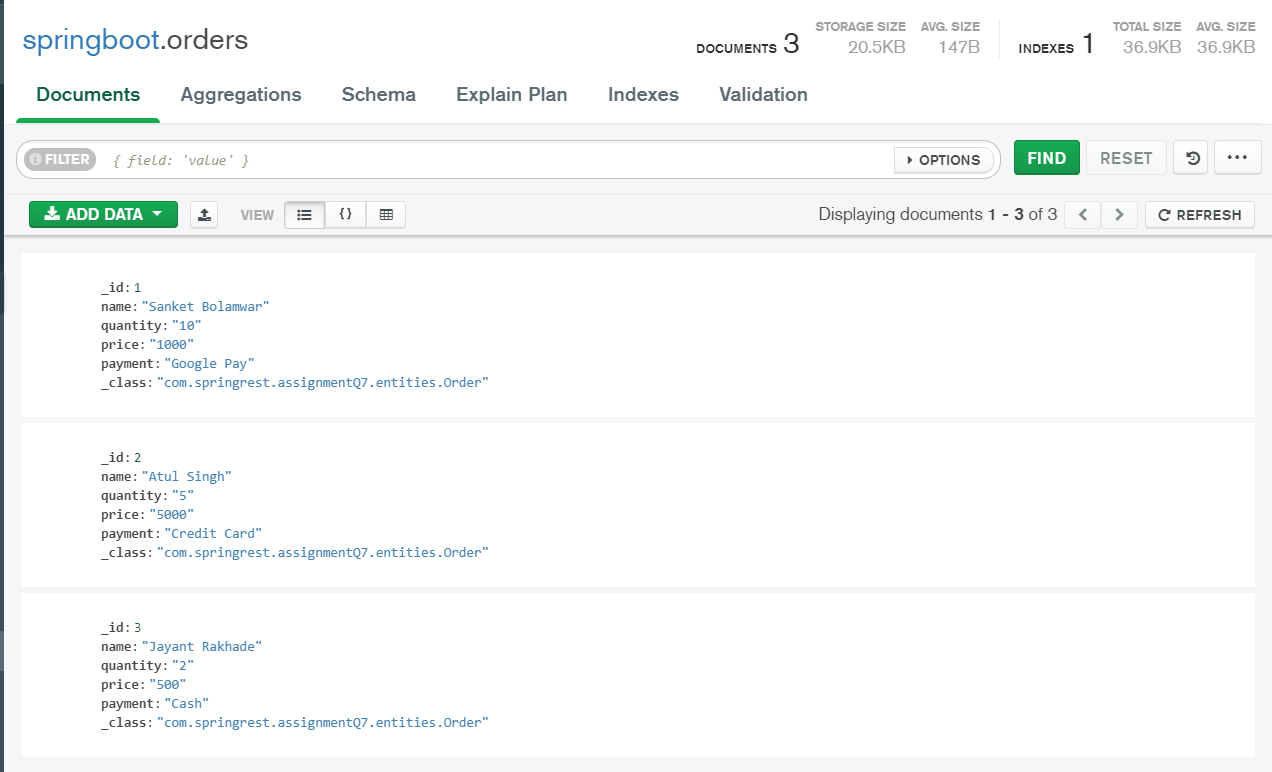
**public** **void** deleteOrder(**int** id) {

orderRepo.deleteById(id);

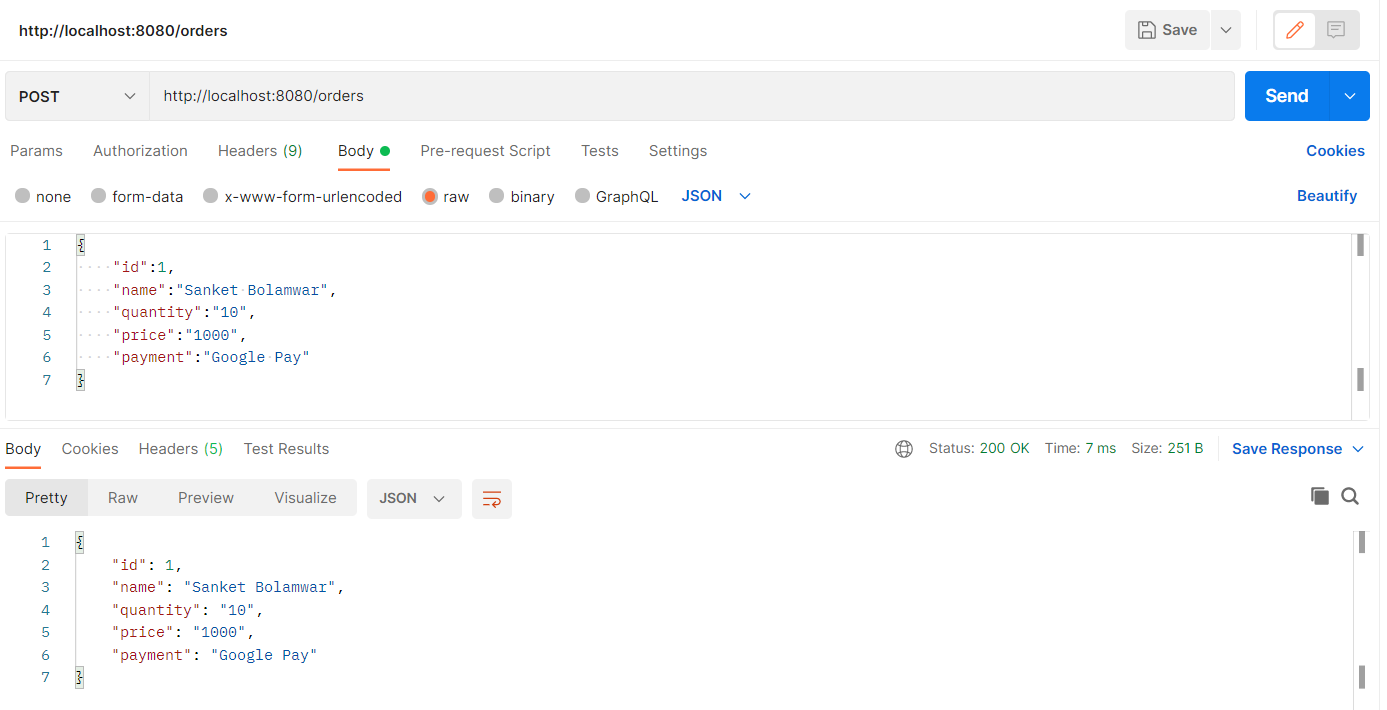
}

}

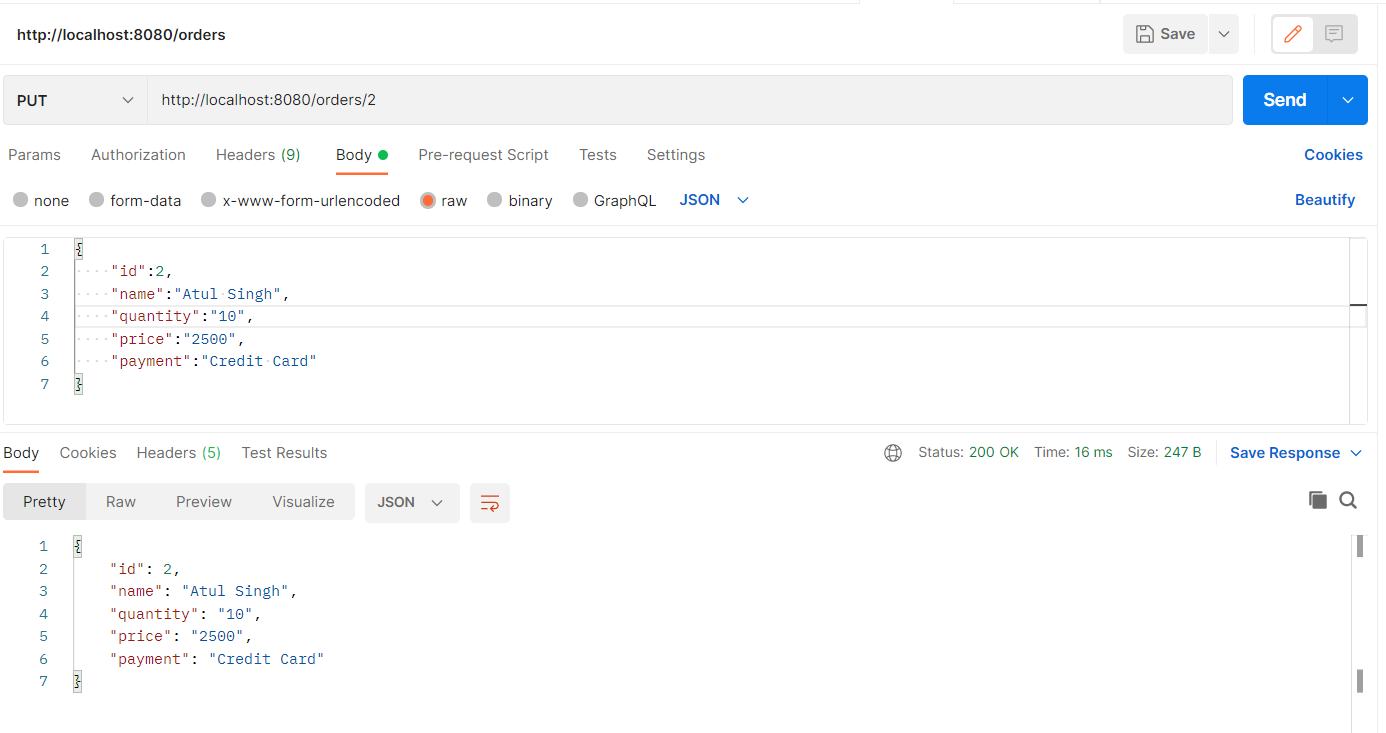
**Database:**

****

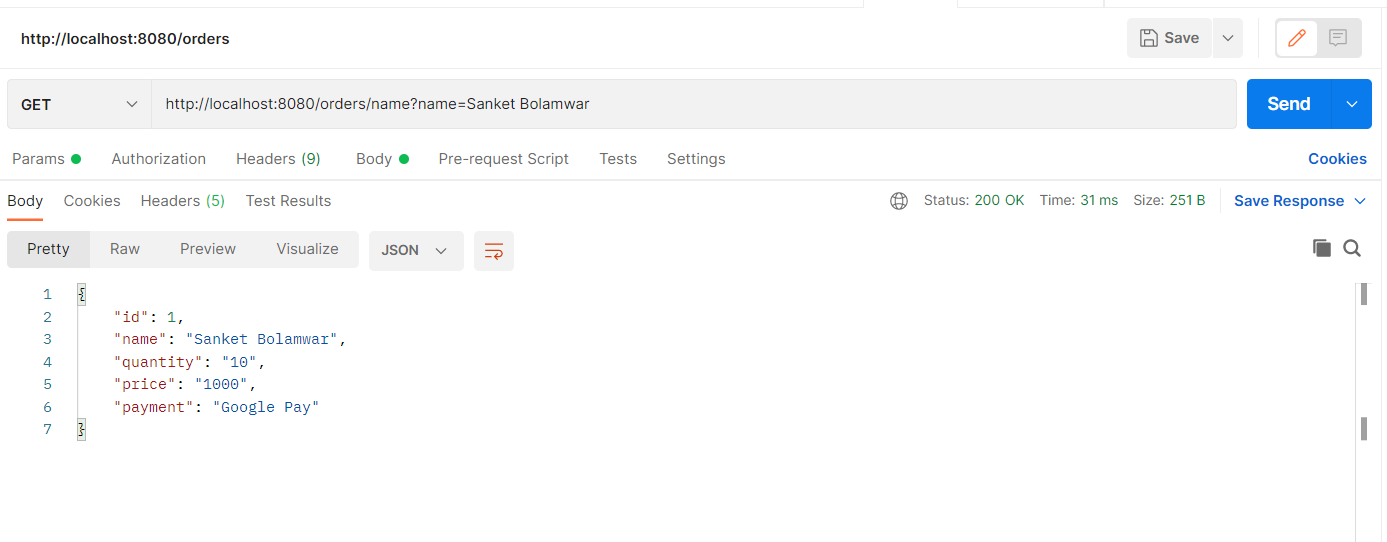
1. **A user can place an order**

****

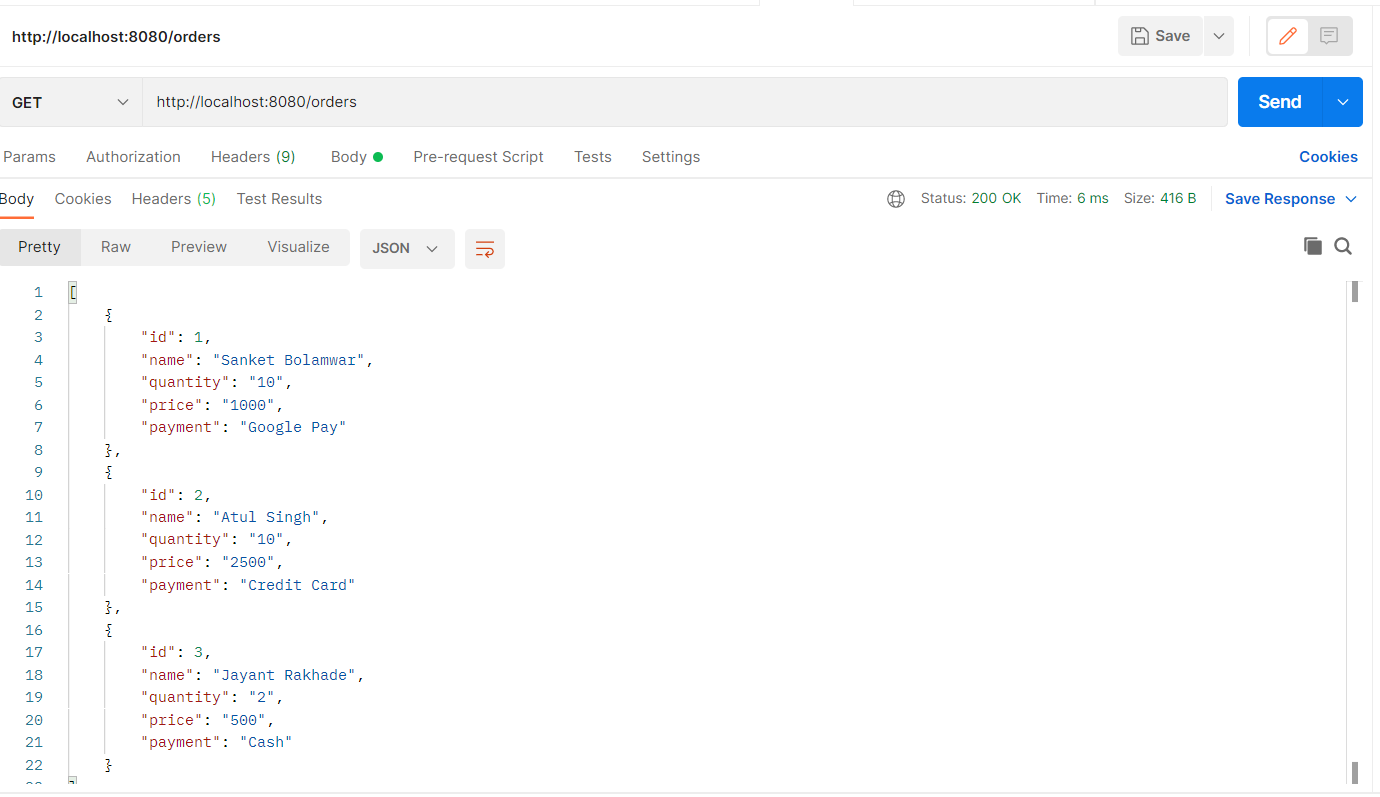
1. **A user can update an order**

****

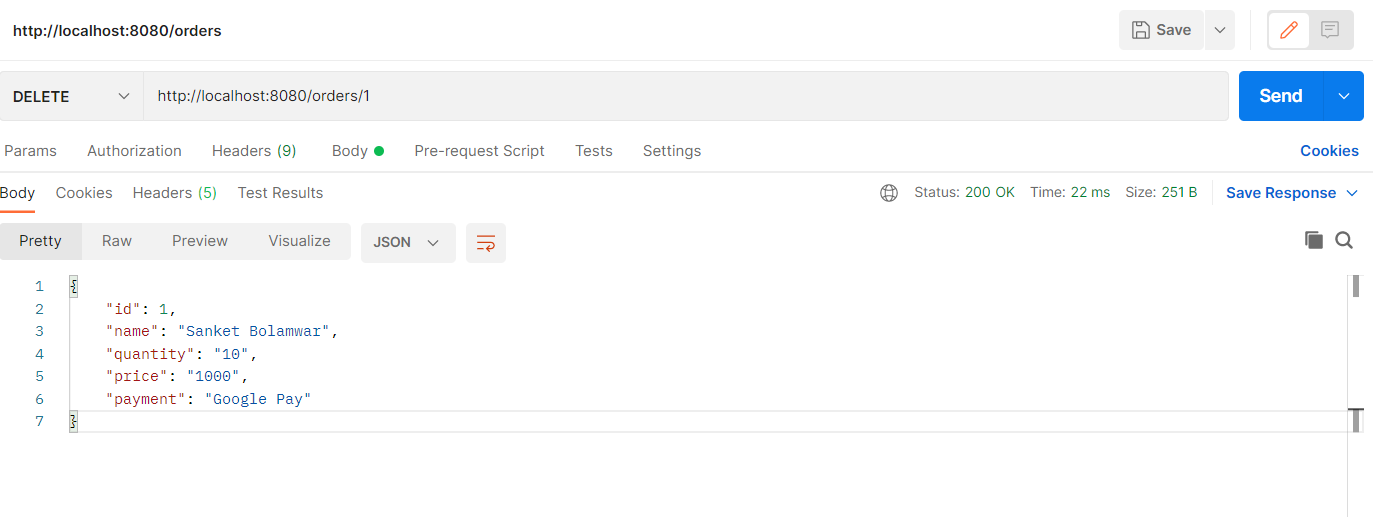
1. **A user can view specific order**

****

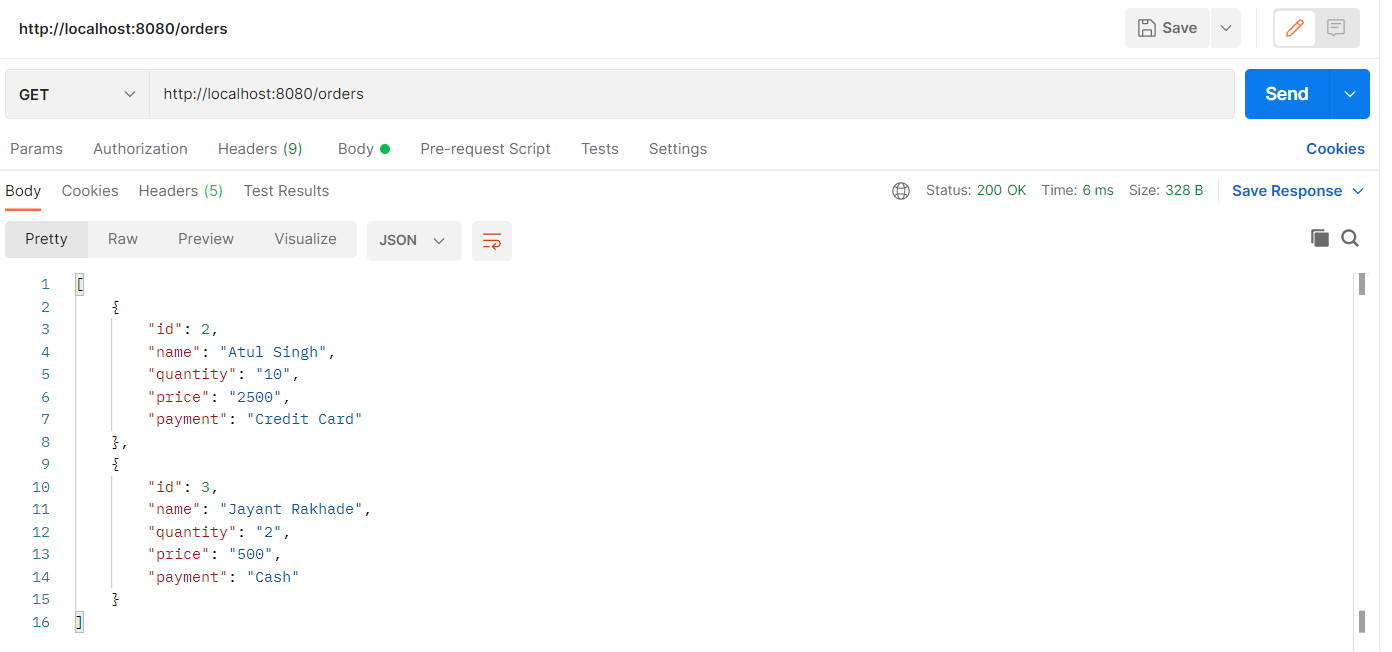
1. **A user can view all the orders**

****

1. **A user can delete a specific order.**

****

**After Delete:**

****

1. Design and develop RESTful web service as follows:
   1. An admin can add a new product.
   2. An admin can update details of existing product.
   3. An admin can delete existing product
   4. An admin can view specific product
   5. An admin can view all the products

Note: Use MongoRepository of Spring data to store product details.

**Refere = Spring.REST.AssignmentQ8**

**// Refer following Product Related Classes files.**

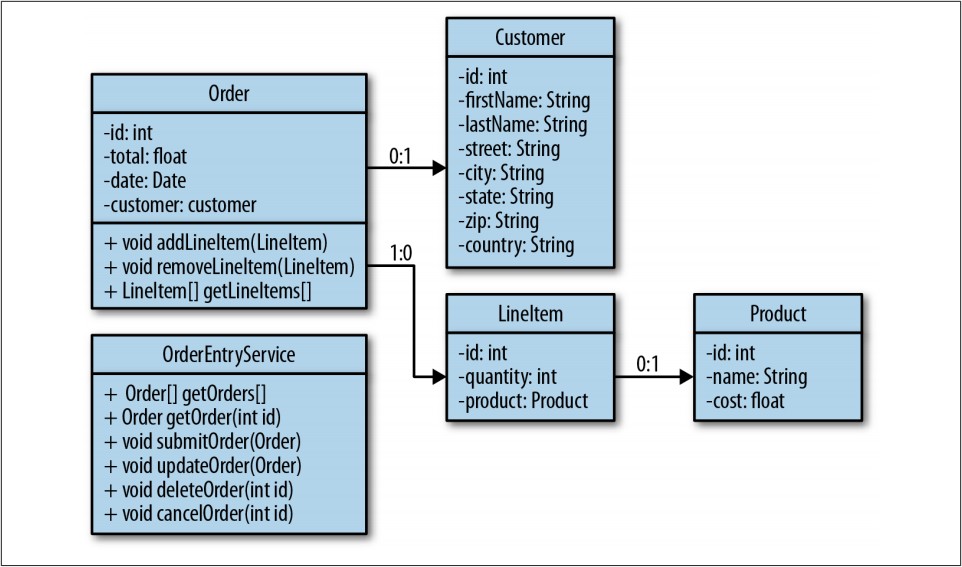
**Application Class:**

**Product Class:**

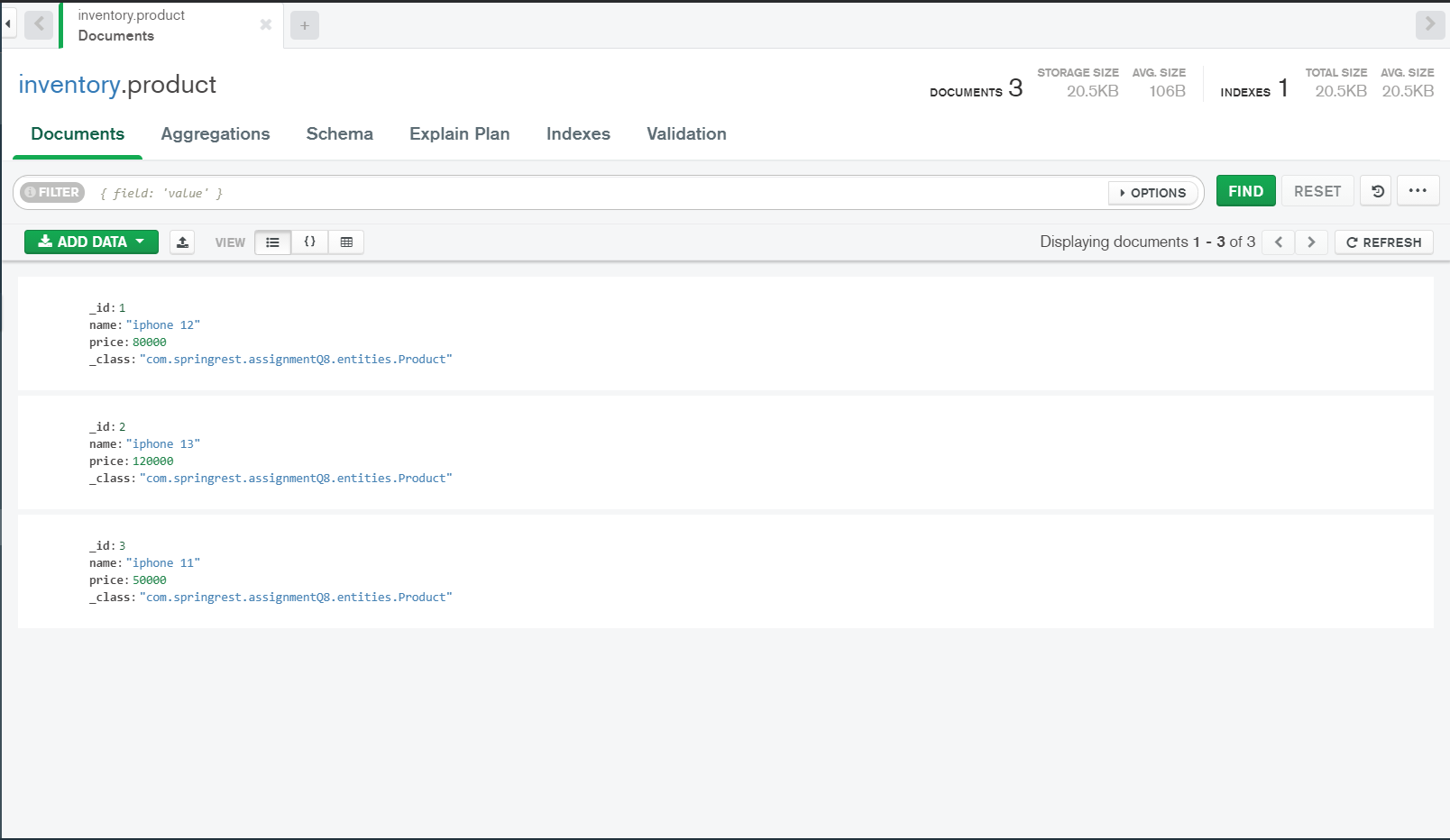
**ProductController:**

**ProductRepo:**

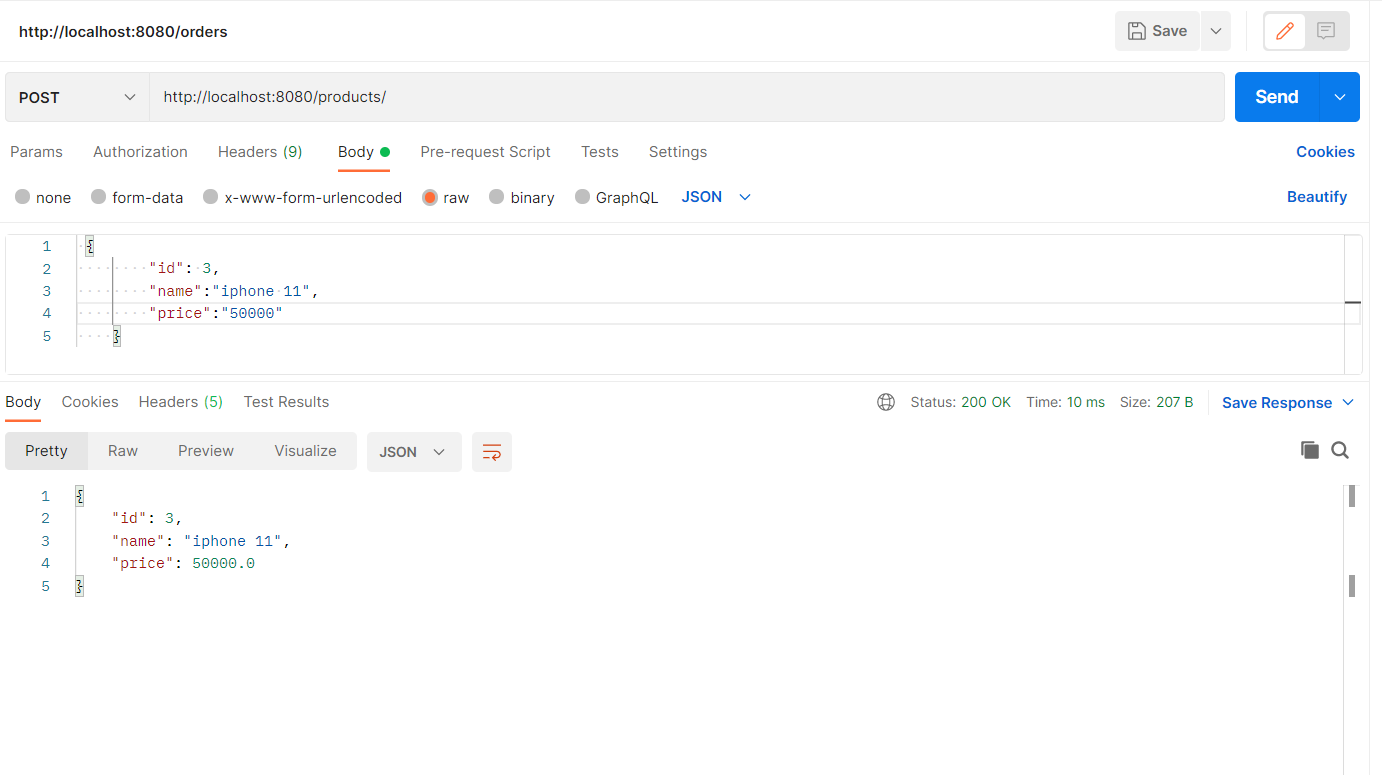
**ProductService:**



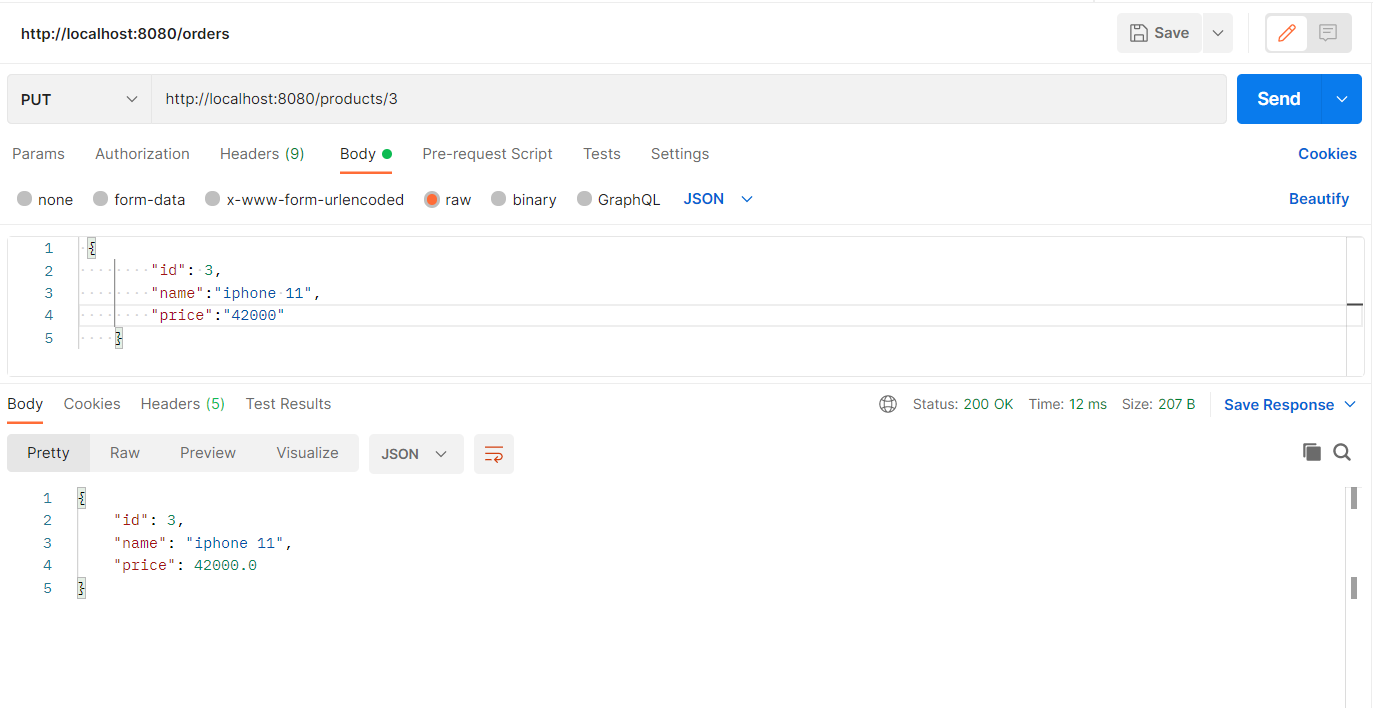
**Database :**

****

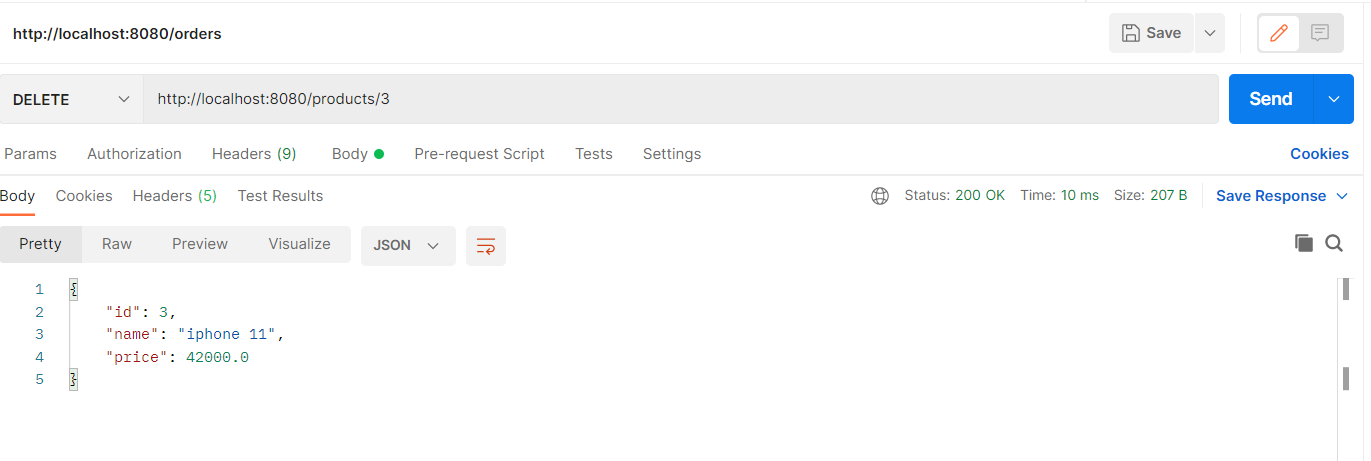
1. **An admin can add a new product.**

****

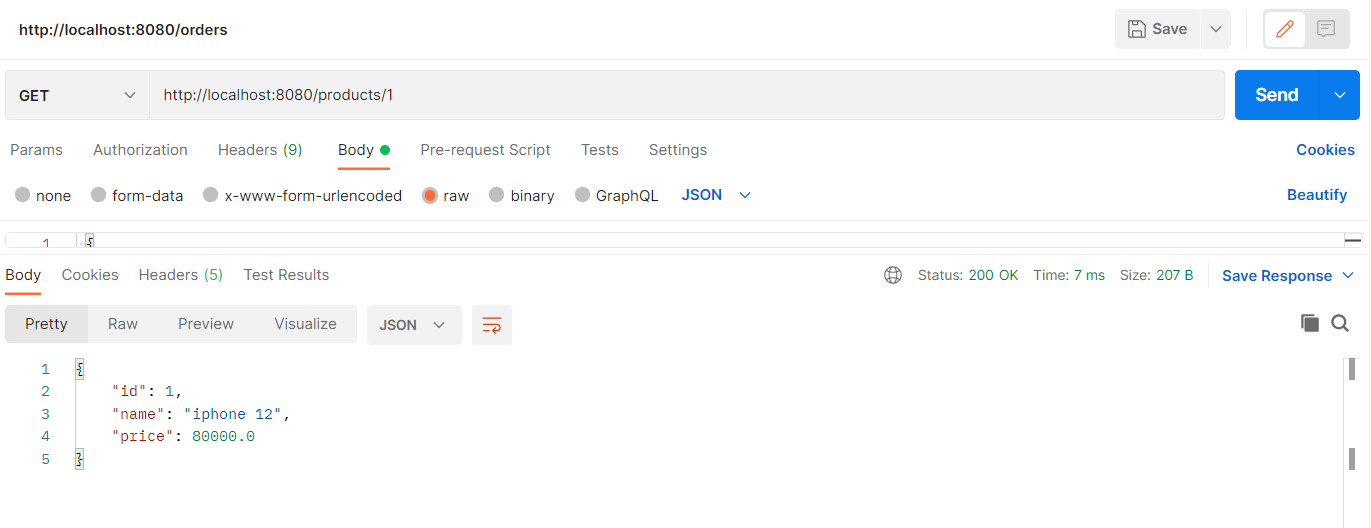
1. **An admin can update details of existing product.**

****

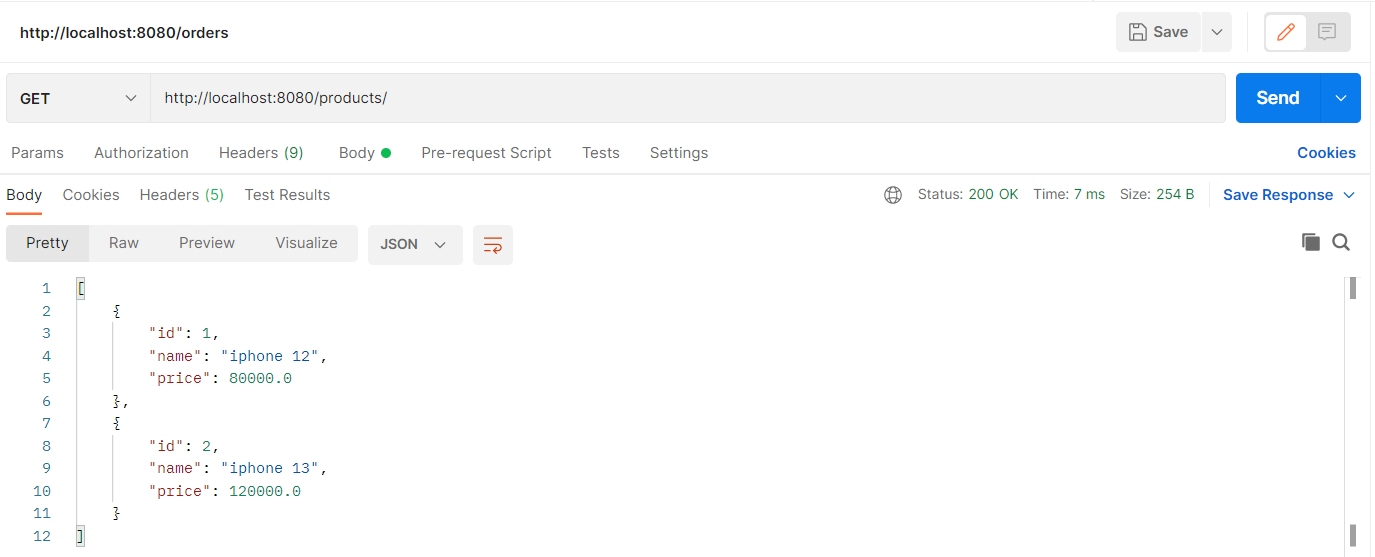
1. **An admin can delete existing product**

****

1. **An admin can view specific product**

****

1. **An admin can view all the products**



1. Design and develop RESTful web service as follows:
   1. Add a new customer information.
   2. Update customer information.
   3. Delete existing customer information.
   4. Fetch information of specific customer.
   5. Fetch information of all customers.

Note: Use CrudRepository of Spring Data to store customer details.

**Refere = Spring.REST.AssignmentQ8**

**// Refer following Customer Related Classes files.**

**Application Class:**

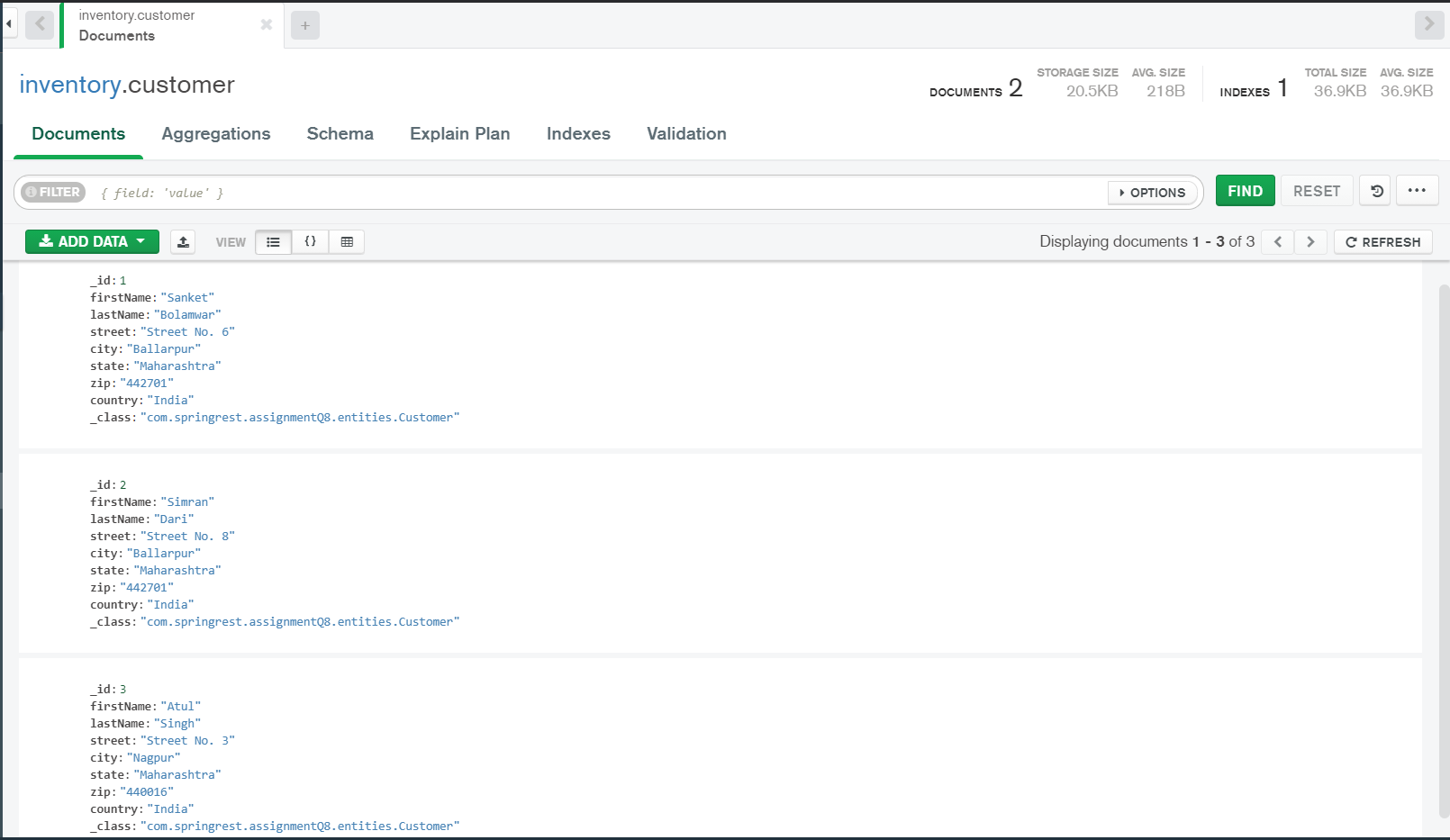
**Customer Class:**

**CustomerController:**

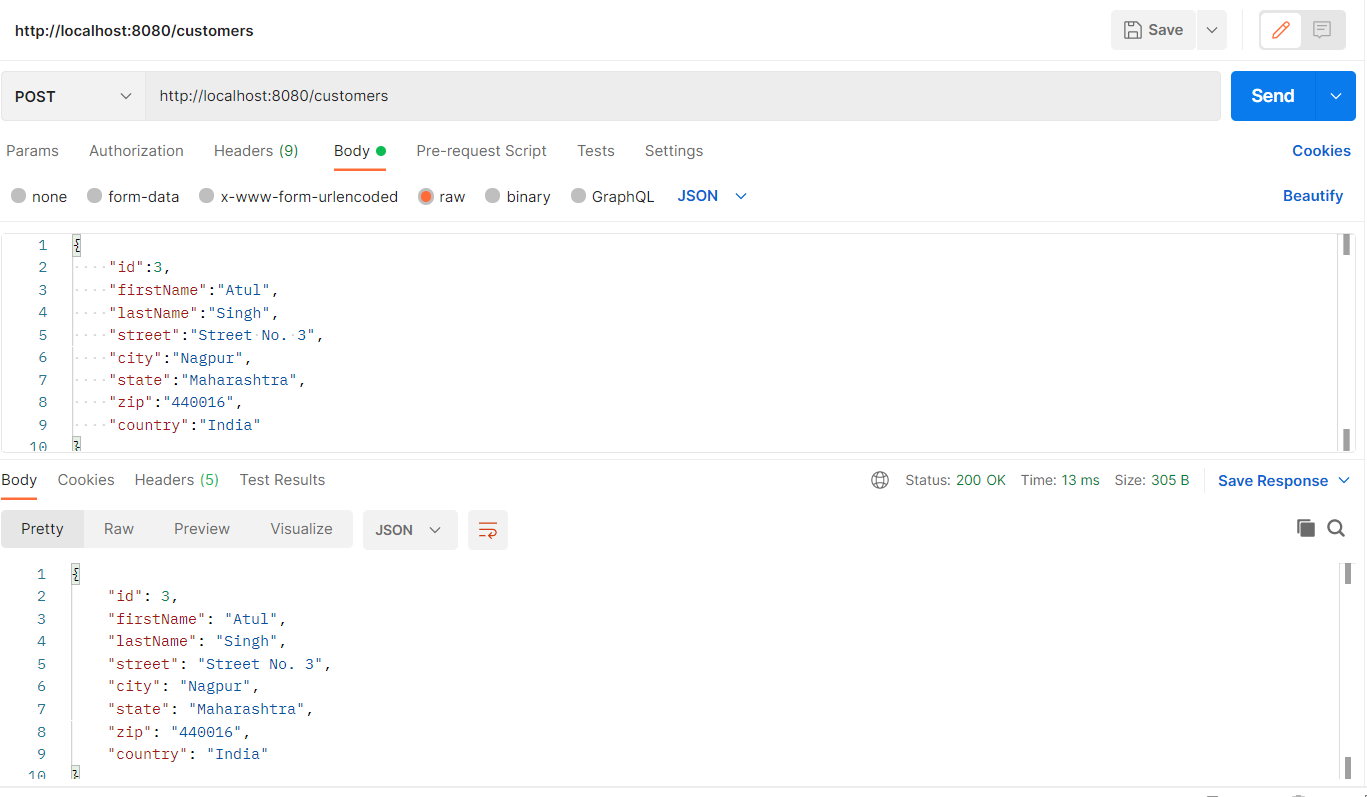
**CustomerRepo:**

**CustomerService:**

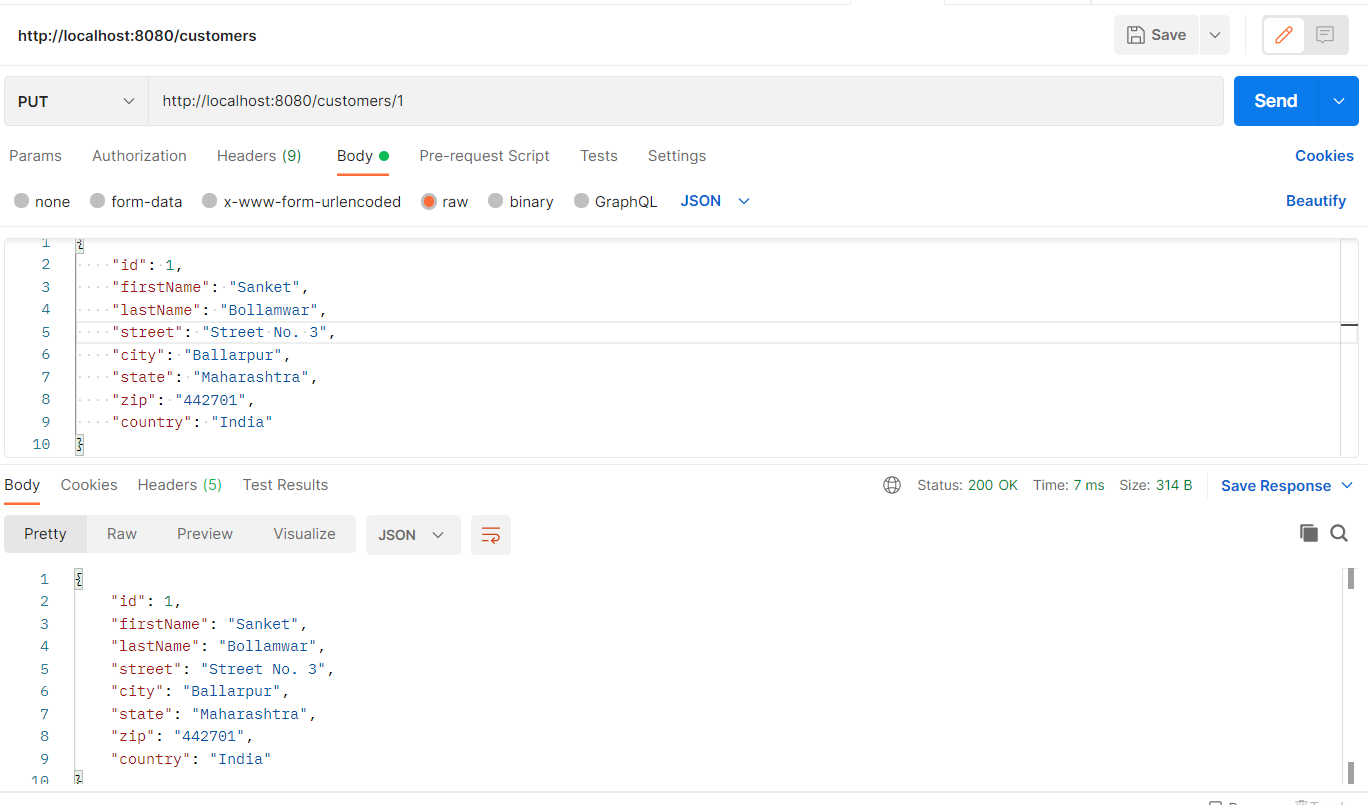
**Database:**

****

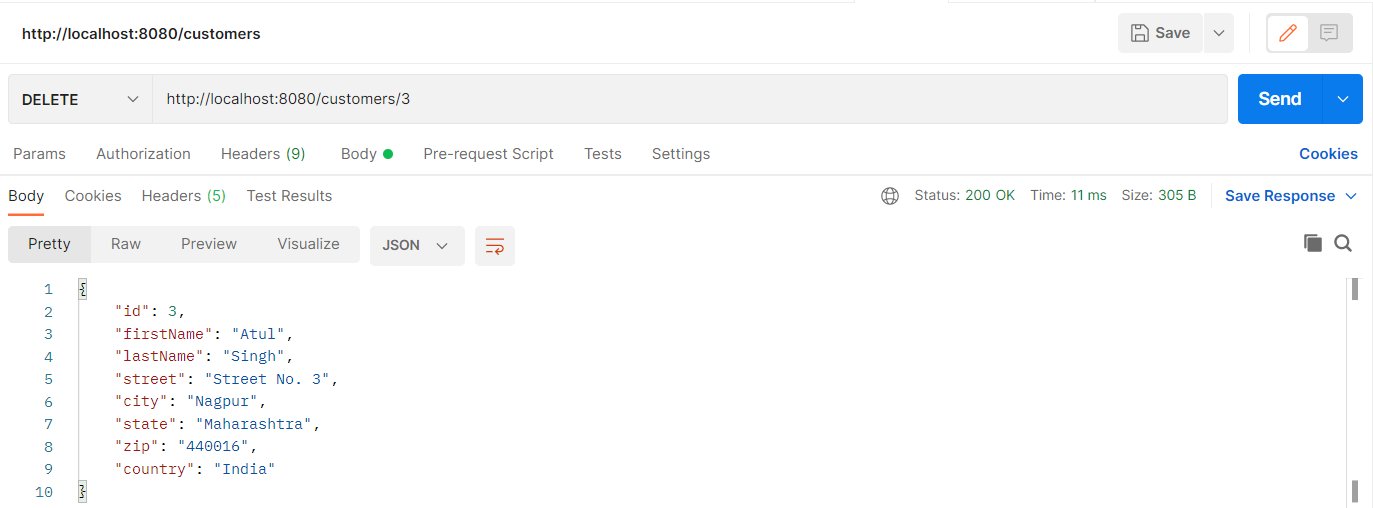
1. **Add a new customer information.**

****

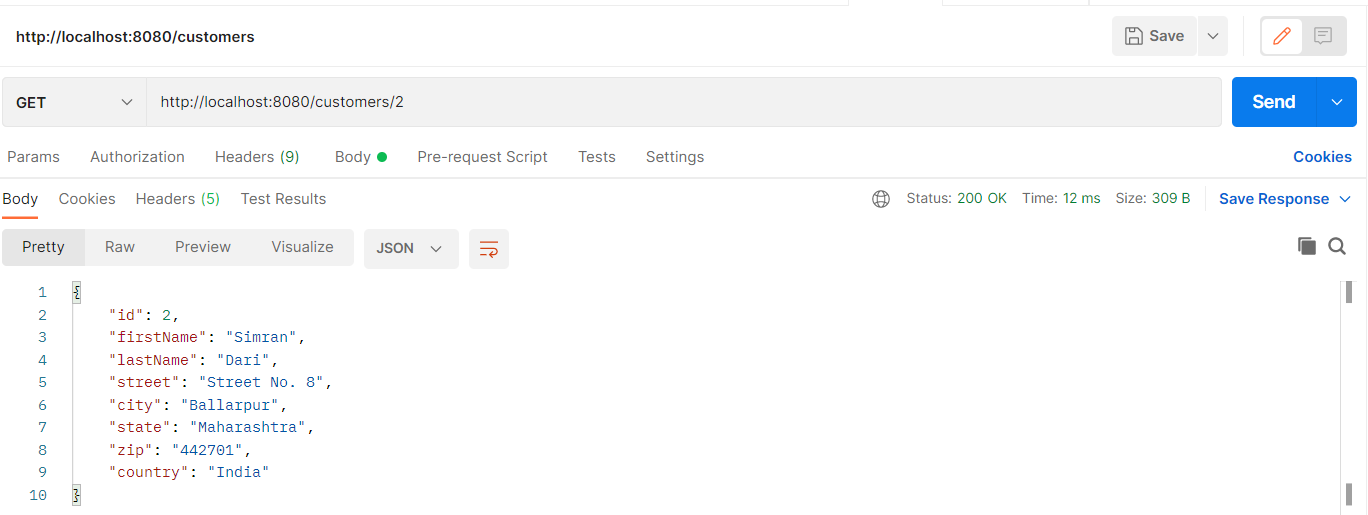
1. **Update customer information.**

****

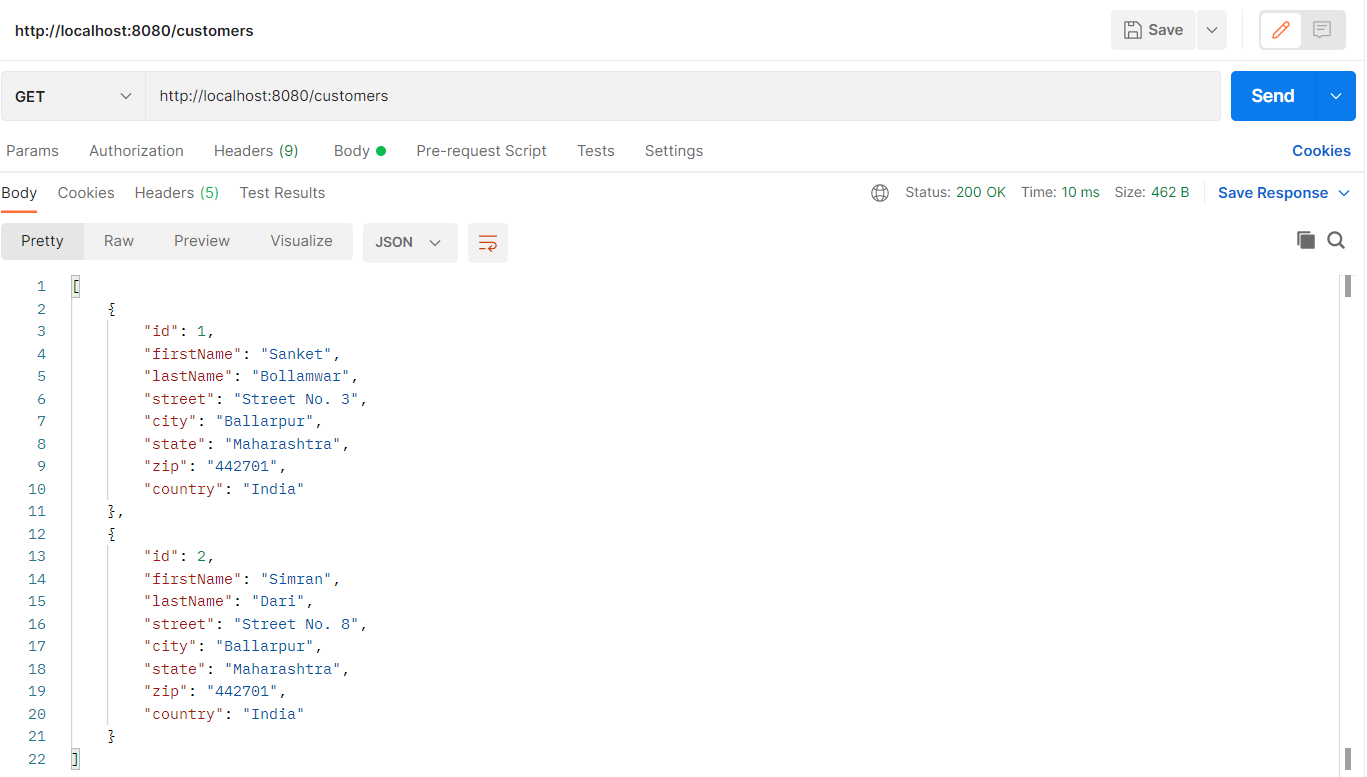
1. **Delete existing customer information.**

****

1. **Fetch information of specific customer.**

****

1. **Fetch information of all customers.**

****

1. Test all the above RESTful web services by using MockMvc.