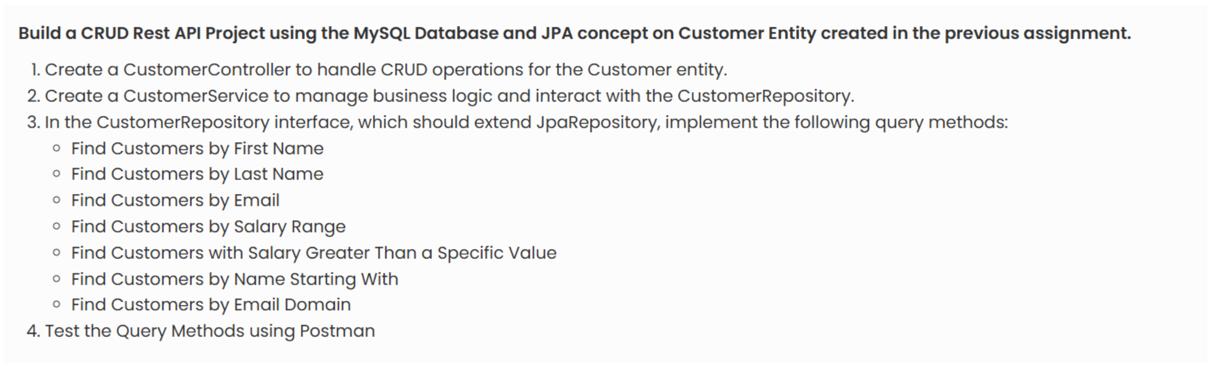
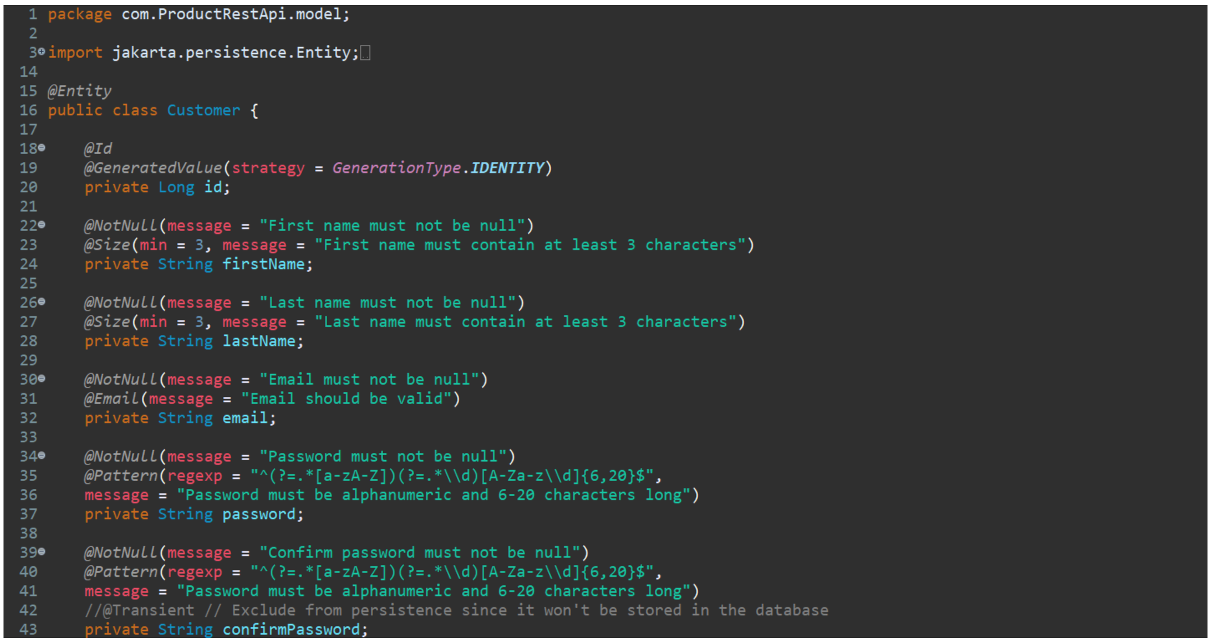


Build a CRUD Rest API Project using the MySQL Database and JPA concept on Customer Entity created in the previous assignment.

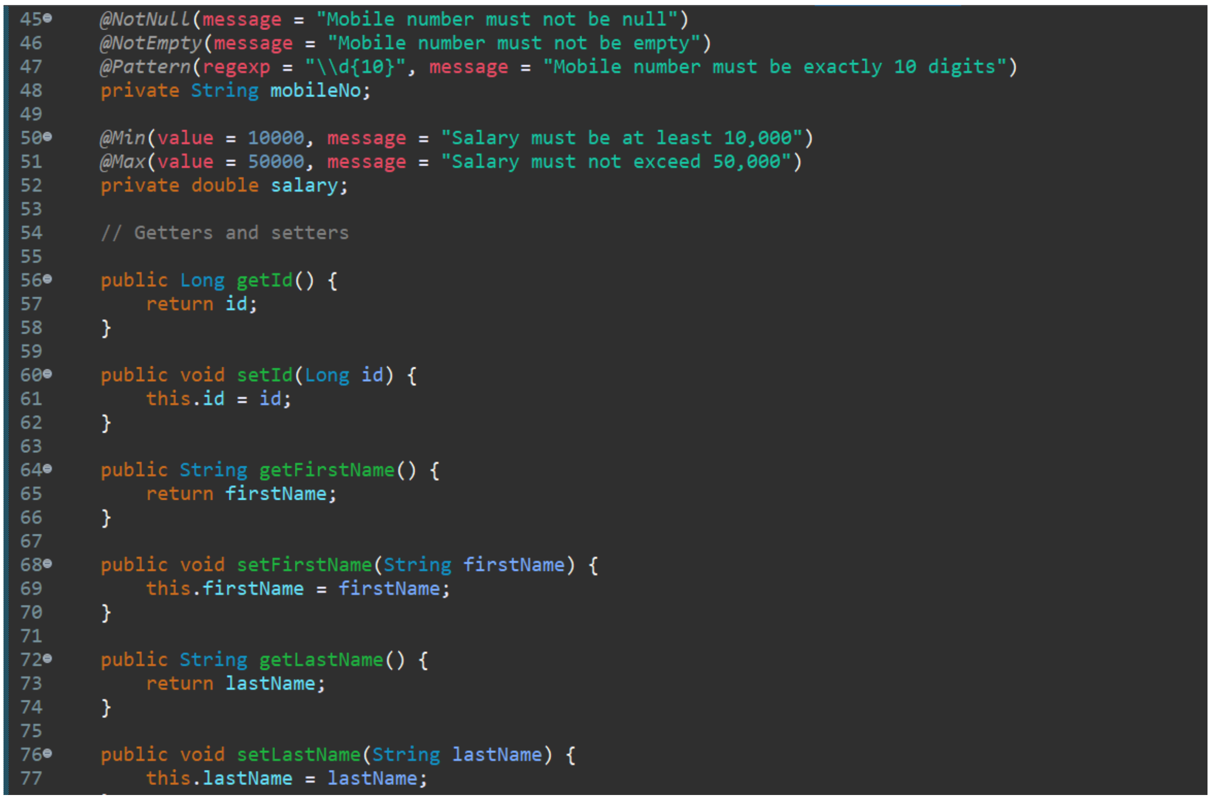
1. Create a CustomerController to handle CRUD operations for the Customer entity.
2. Create a CustomerService to manage business logic and interact with the CustomerRepository.
3. In the CustomerRepository interface, which should extend JpaRepository, implement the following query methods:
    - Find Customers by First Name
    - Find Customers by Last Name
    - Find Customers by Email
    - Find Customers by Salary Range
    - Find Customers with Salary Greater Than a Specific Value
    - Find Customers by Name Starting With
    - Find Customers by Email Domain
4. Test the Query Methods using Postman

Code-

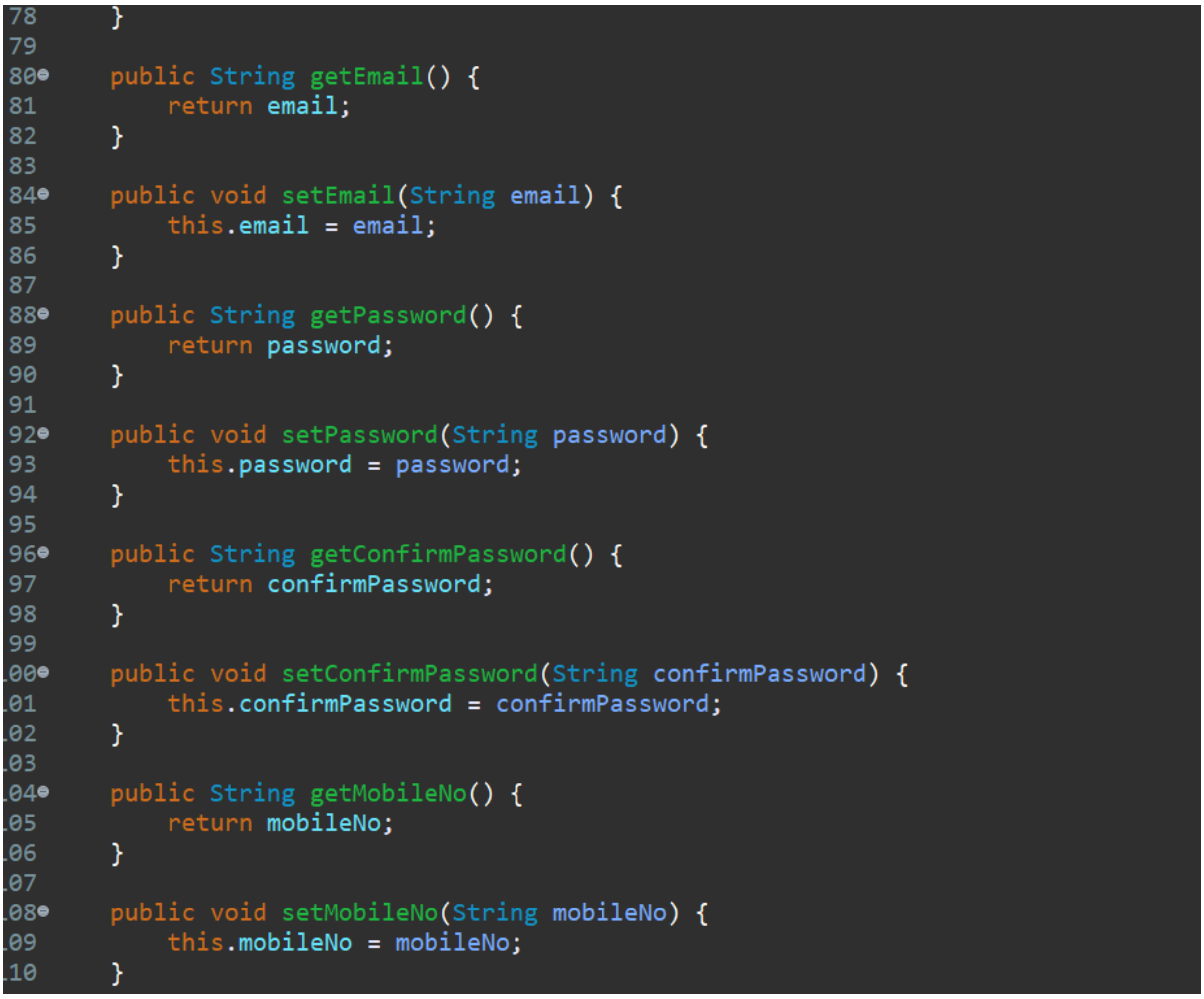
Customer.java (Entity)



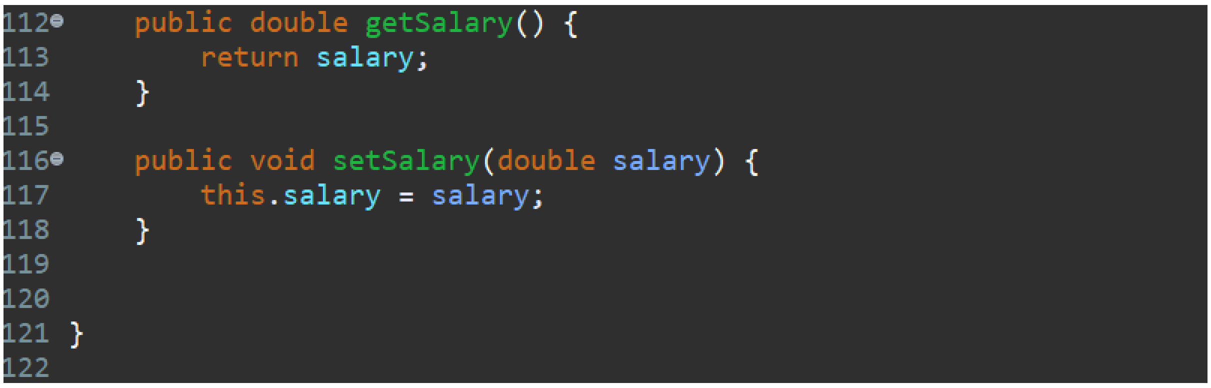
```java
package com.ProductRestApi.model;

import jakarta.persistence.Entity;

@Entity
public class Customer {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @NotNull(message = "First name must not be null")
    @Size(min = 3, message = "First name must contain at least 3 characters")
    private String firstName;

    @NotNull(message = "Last name must not be null")
    @Size(min = 3, message = "Last name must contain at least 3 characters")
    private String lastName;

    @NotNull(message = "Email must not be null")
    @Email(message = "Email should be valid")
    private String email;

    @NotNull(message = "Password must not be null")
    @Pattern(regexp = "^(?=.*[a-zA-Z])(?=.*\\d)[A-Za-z\\d]{6,20}$",
    message = "Password must be alphanumeric and 6-20 characters long")
    private String password;

    @NotNull(message = "Confirm password must not be null")
    @Pattern(regexp = "^(?=.*[a-zA-Z])(?=.*\\d)[A-Za-z\\d]{6,20}$",
    message = "Password must be alphanumeric and 6-20 characters long")
    //@Transient // Exclude from persistence since it won't be stored in the database
    private String confirmPassword;
```



```java
    @NotNull(message = "Mobile number must not be null")
    @NotEmpty(message = "Mobile number must not be empty")
    @Pattern(regexp = "\\d{10}", message = "Mobile number must be exactly 10 digits")
    private String mobileNo;

    @Min(value = 10000, message = "Salary must be at least 10,000")
    @Max(value = 50000, message = "Salary must not exceed 50,000")
    private double salary;

    // Getters and setters

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
```

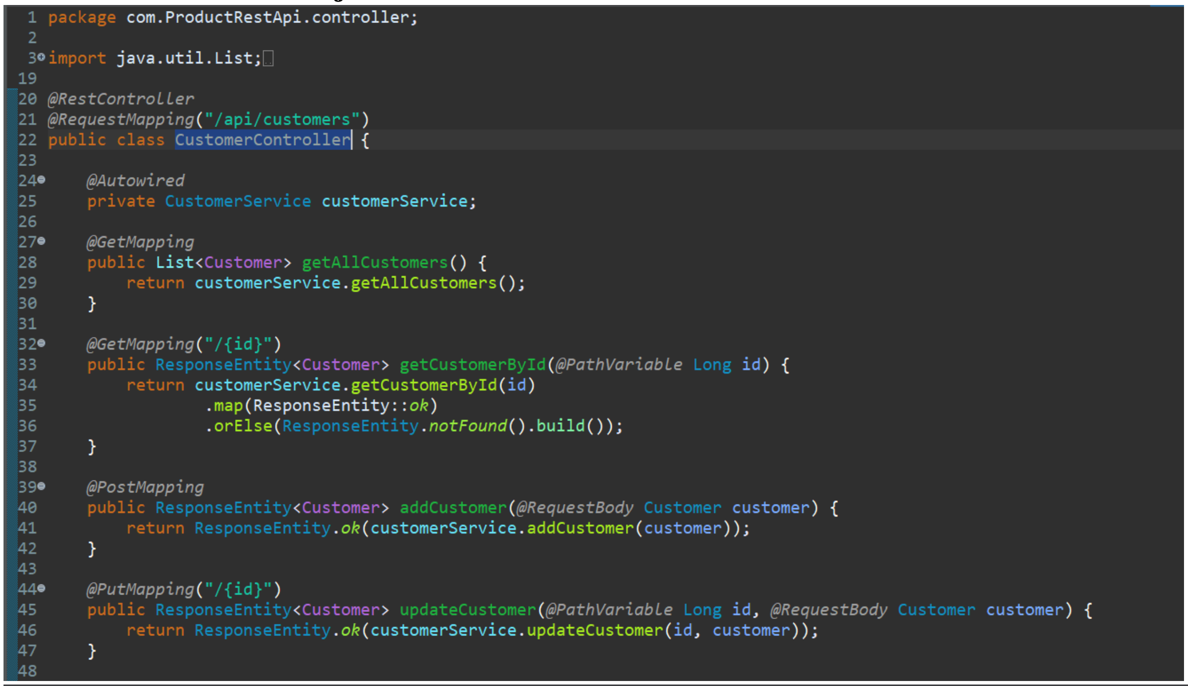


```java
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getConfirmPassword() {
        return confirmPassword;
    }

    public void setConfirmPassword(String confirmPassword) {
        this.confirmPassword = confirmPassword;
    }

    public String getMobileNo() {
        return mobileNo;
    }

    public void setMobileNo(String mobileNo) {
        this.mobileNo = mobileNo;
    }
```

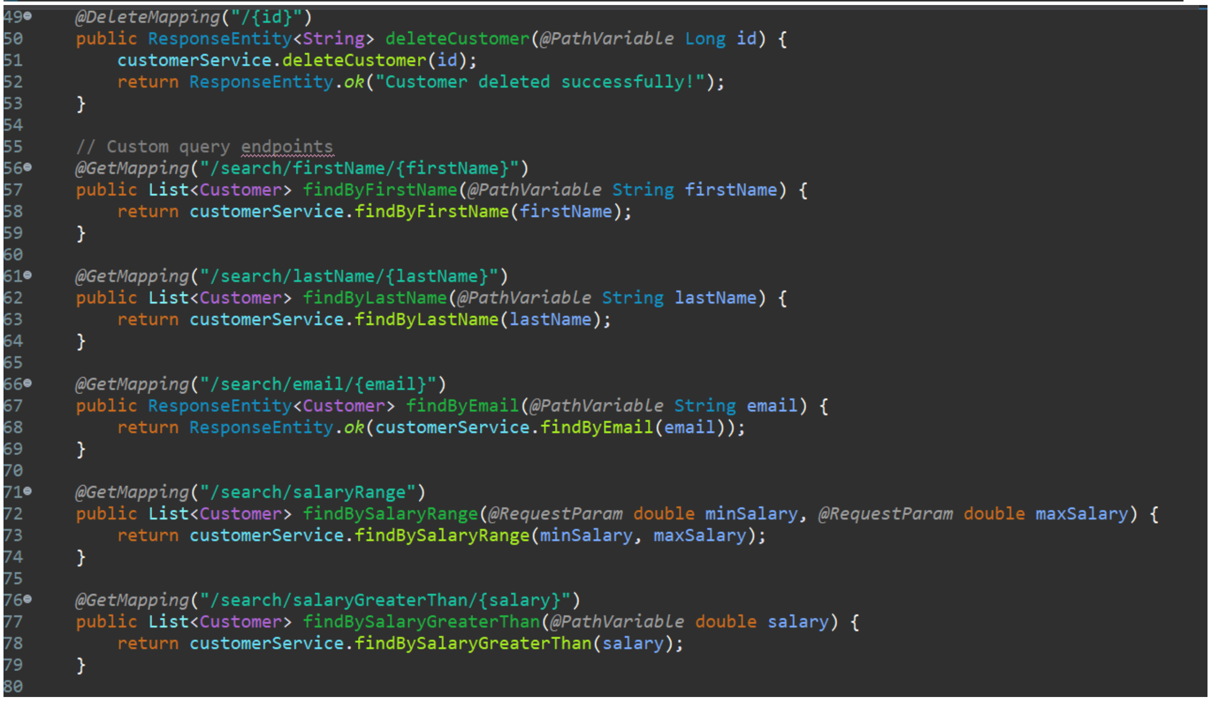


```java
    public double getSalary() {
        return salary;
    }

    public void setSalary(double salary) {
        this.salary = salary;
    }


}

```

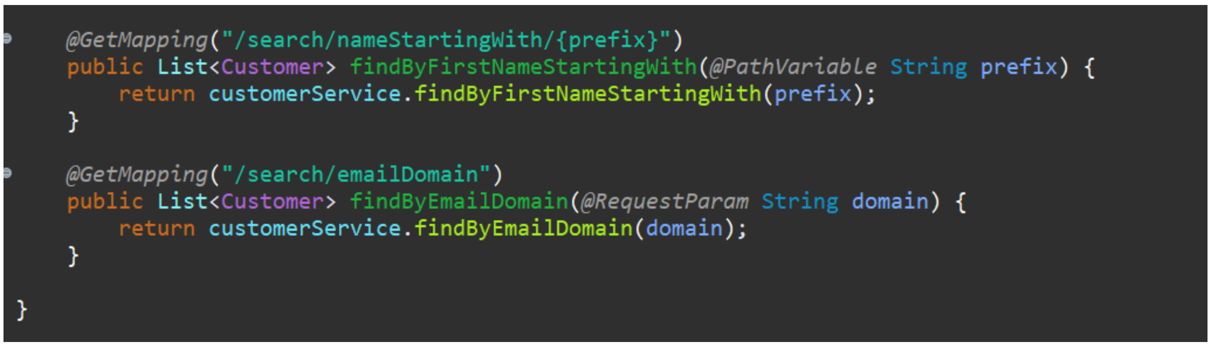
CustomerController.java



```java
package com.ProductRestApi.controller;

import java.util.List;

@RestController
@RequestMapping("/api/customers")
public class CustomerController {

    @Autowired
    private CustomerService customerService;

    @GetMapping
    public List<Customer> getAllCustomers() {
        return customerService.getAllCustomers();
    }

    @GetMapping("/{id}")
    public ResponseEntity<Customer> getCustomerById(@PathVariable Long id) {
        return customerService.getCustomerById(id)
                .map(ResponseEntity::ok)
                .orElse(ResponseEntity.notFound().build());
    }

    @PostMapping
    public ResponseEntity<Customer> addCustomer(@RequestBody Customer customer) {
        return ResponseEntity.ok(customerService.addCustomer(customer));
    }

    @PutMapping("/{id}")
    public ResponseEntity<Customer> updateCustomer(@PathVariable Long id, @RequestBody Customer customer) {
        return ResponseEntity.ok(customerService.updateCustomer(id, customer));
    }

```

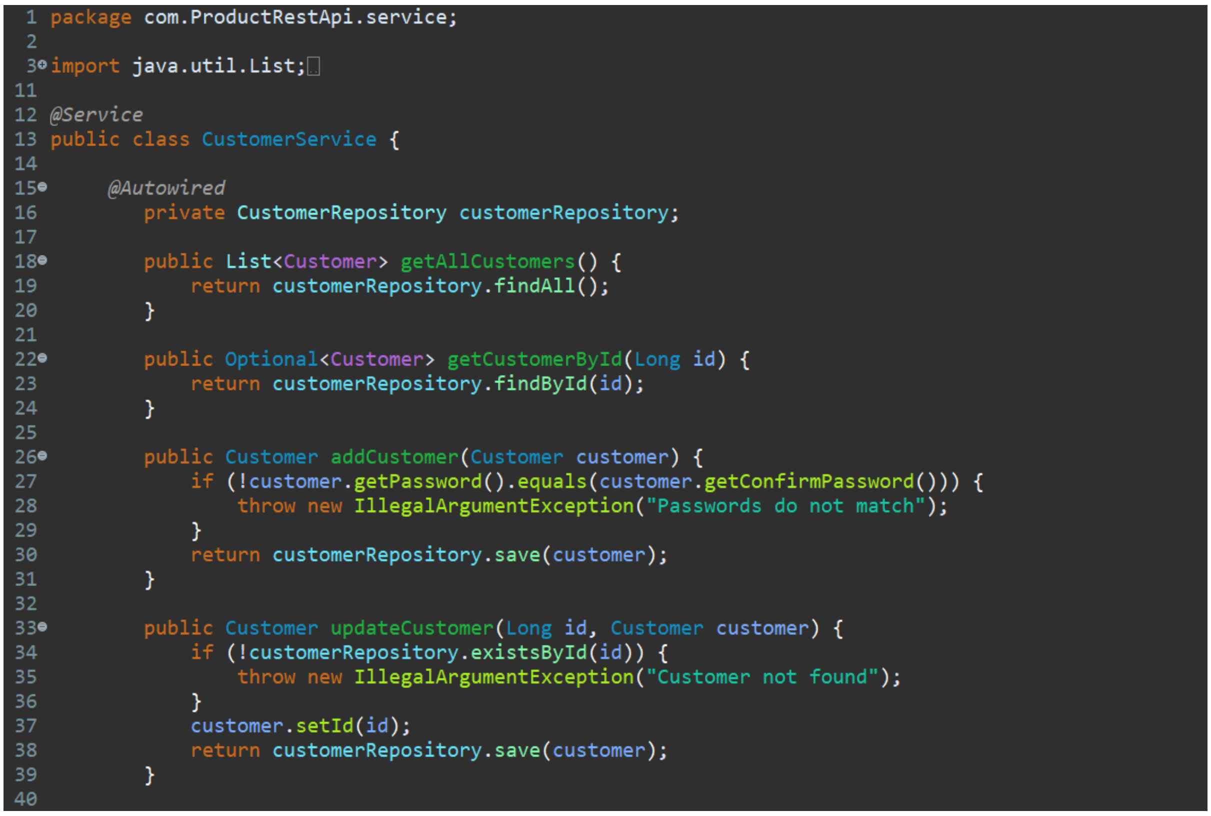


```java
    @DeleteMapping("/{id}")
    public ResponseEntity<String> deleteCustomer(@PathVariable Long id) {
        customerService.deleteCustomer(id);
        return ResponseEntity.ok("Customer deleted successfully!");
    }

    // Custom query endpoints
    @GetMapping("/search/firstName/{firstName}")
    public List<Customer> findByFirstName(@PathVariable String firstName) {
        return customerService.findByFirstName(firstName);
    }

    @GetMapping("/search/lastName/{lastName}")
    public List<Customer> findByLastName(@PathVariable String lastName) {
        return customerService.findByLastName(lastName);
    }

    @GetMapping("/search/email/{email}")
    public ResponseEntity<Customer> findByEmail(@PathVariable String email) {
        return ResponseEntity.ok(customerService.findByEmail(email));
    }

    @GetMapping("/search/salaryRange")
    public List<Customer> findBySalaryRange(@RequestParam double minSalary, @RequestParam double maxSalary) {
        return customerService.findBySalaryRange(minSalary, maxSalary);
    }

    @GetMapping("/search/salaryGreaterThan/{salary}")
    public List<Customer> findBySalaryGreaterThan(@PathVariable double salary) {
        return customerService.findBySalaryGreaterThan(salary);
    }

```

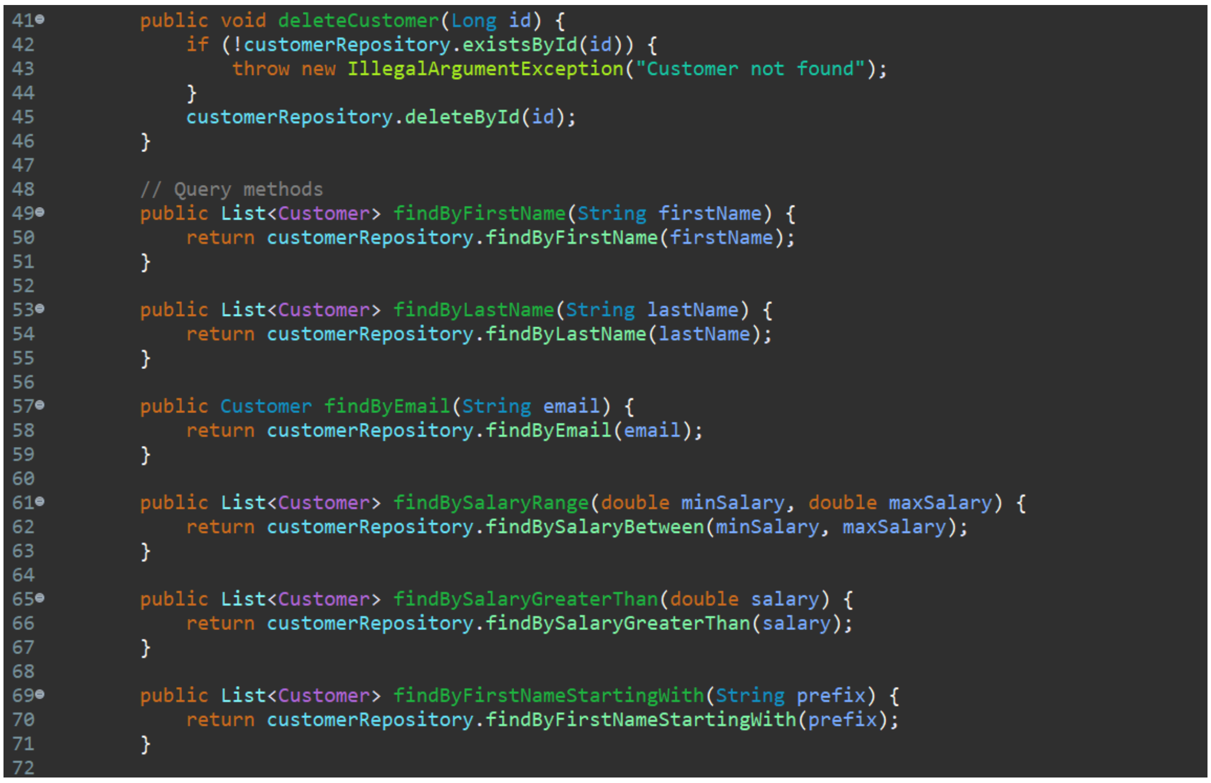


```java
    @GetMapping("/search/nameStartingWith/{prefix}")
    public List<Customer> findByFirstNameStartingWith(@PathVariable String prefix) {
        return customerService.findByFirstNameStartingWith(prefix);
    }

    @GetMapping("/search/emailDomain")
    public List<Customer> findByEmailDomain(@RequestParam String domain) {
        return customerService.findByEmailDomain(domain);
    }

}
```

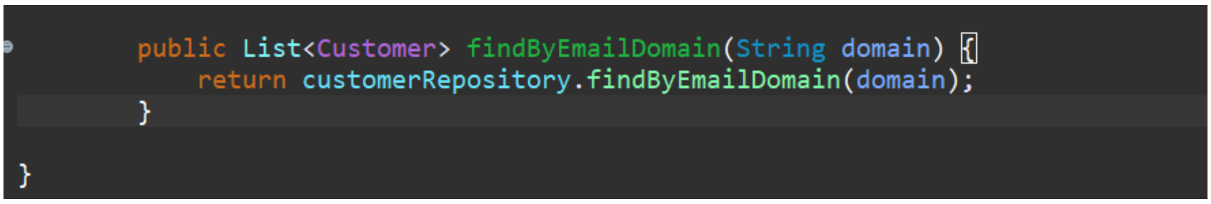
CustomerService.java



```java
package com.ProductRestApi.service;

import java.util.List;

@Service
public class CustomerService {

    @Autowired
        private CustomerRepository customerRepository;

        public List<Customer> getAllCustomers() {
            return customerRepository.findAll();
        }

        public Optional<Customer> getCustomerById(Long id) {
            return customerRepository.findById(id);
        }

        public Customer addCustomer(Customer customer) {
            if (!customer.getPassword().equals(customer.getConfirmPassword())) {
                throw new IllegalArgumentException("Passwords do not match");
            }
            return customerRepository.save(customer);
        }

        public Customer updateCustomer(Long id, Customer customer) {
            if (!customerRepository.existsById(id)) {
                throw new IllegalArgumentException("Customer not found");
            }
            customer.setId(id);
            return customerRepository.save(customer);
        }

```

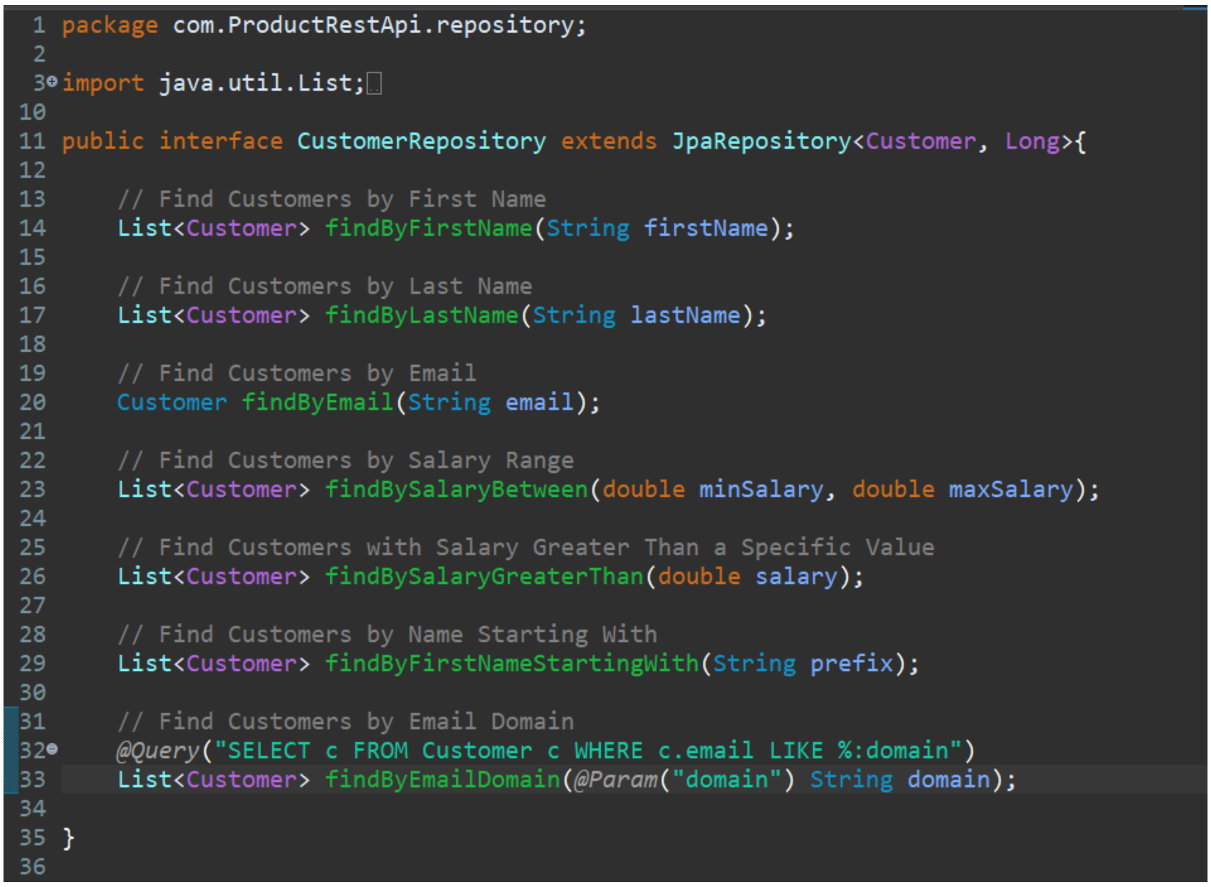


```java
    public void deleteCustomer(Long id) {
        if (!customerRepository.existsById(id)) {
            throw new IllegalArgumentException("Customer not found");
        }
        customerRepository.deleteById(id);
    }

    // Query methods
    public List<Customer> findByFirstName(String firstName) {
        return customerRepository.findByFirstName(firstName);
    }

    public List<Customer> findByLastName(String lastName) {
        return customerRepository.findByLastName(lastName);
    }

    public Customer findByEmail(String email) {
        return customerRepository.findByEmail(email);
    }

    public List<Customer> findBySalaryRange(double minSalary, double maxSalary) {
        return customerRepository.findBySalaryBetween(minSalary, maxSalary);
    }

    public List<Customer> findBySalaryGreaterThan(double salary) {
        return customerRepository.findBySalaryGreaterThan(salary);
    }

    public List<Customer> findByFirstNameStartingWith(String prefix) {
        return customerRepository.findByFirstNameStartingWith(prefix);
    }

```



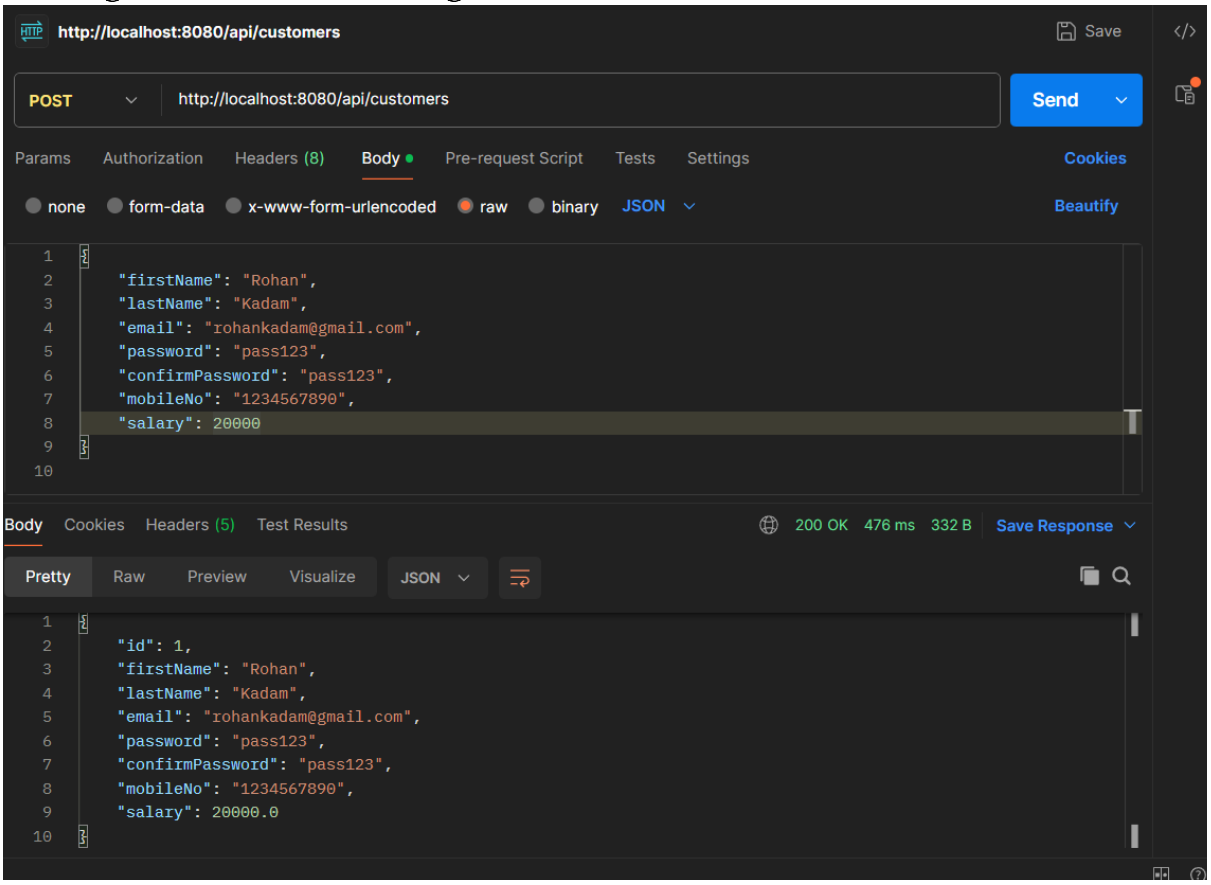
```java
    public List<Customer> findByEmailDomain(String domain) {
        return customerRepository.findByEmailDomain(domain);
    }

}
```

CustomerRepository.java



```java
package com.ProductRestApi.repository;

import java.util.List;

public interface CustomerRepository extends JpaRepository<Customer, Long>{

    // Find Customers by First Name
    List<Customer> findByFirstName(String firstName);

    // Find Customers by Last Name
    List<Customer> findByLastName(String lastName);

    // Find Customers by Email
    Customer findByEmail(String email);

    // Find Customers by Salary Range
    List<Customer> findBySalaryBetween(double minSalary, double maxSalary);

    // Find Customers with Salary Greater Than a Specific Value
    List<Customer> findBySalaryGreaterThan(double salary);

    // Find Customers by Name Starting With
    List<Customer> findByFirstNameStartingWith(String prefix);

    // Find Customers by Email Domain
    @Query("SELECT c FROM Customer c WHERE c.email LIKE %:domain")
    List<Customer> findByEmailDomain(@Param("domain") String domain);

}
```

Adding the Customer through Postman

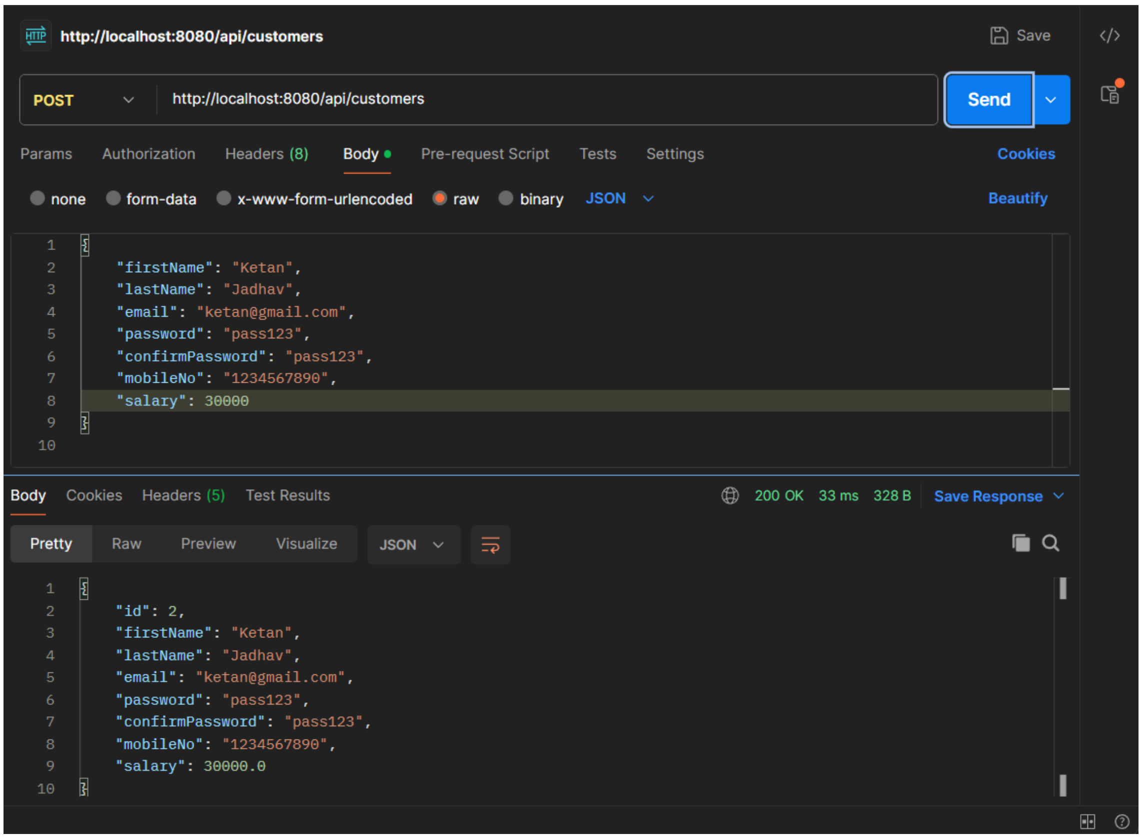


POST http://localhost:8080/api/customers

```
{
    "firstName": "Rohan",
    "lastName": "Kadam",
    "email": "rohankadam@gmail.com",
    "password": "pass123",
    "confirmPassword": "pass123",
    "mobileNo": "1234567890",
    "salary": 20000
}
```

200 OK 476 ms 332 B

```
{
    "id": 1,
    "firstName": "Rohan",
    "lastName": "Kadam",
    "email": "rohankadam@gmail.com",
    "password": "pass123",
    "confirmPassword": "pass123",
    "mobileNo": "1234567890",
    "salary": 20000.0
}
```



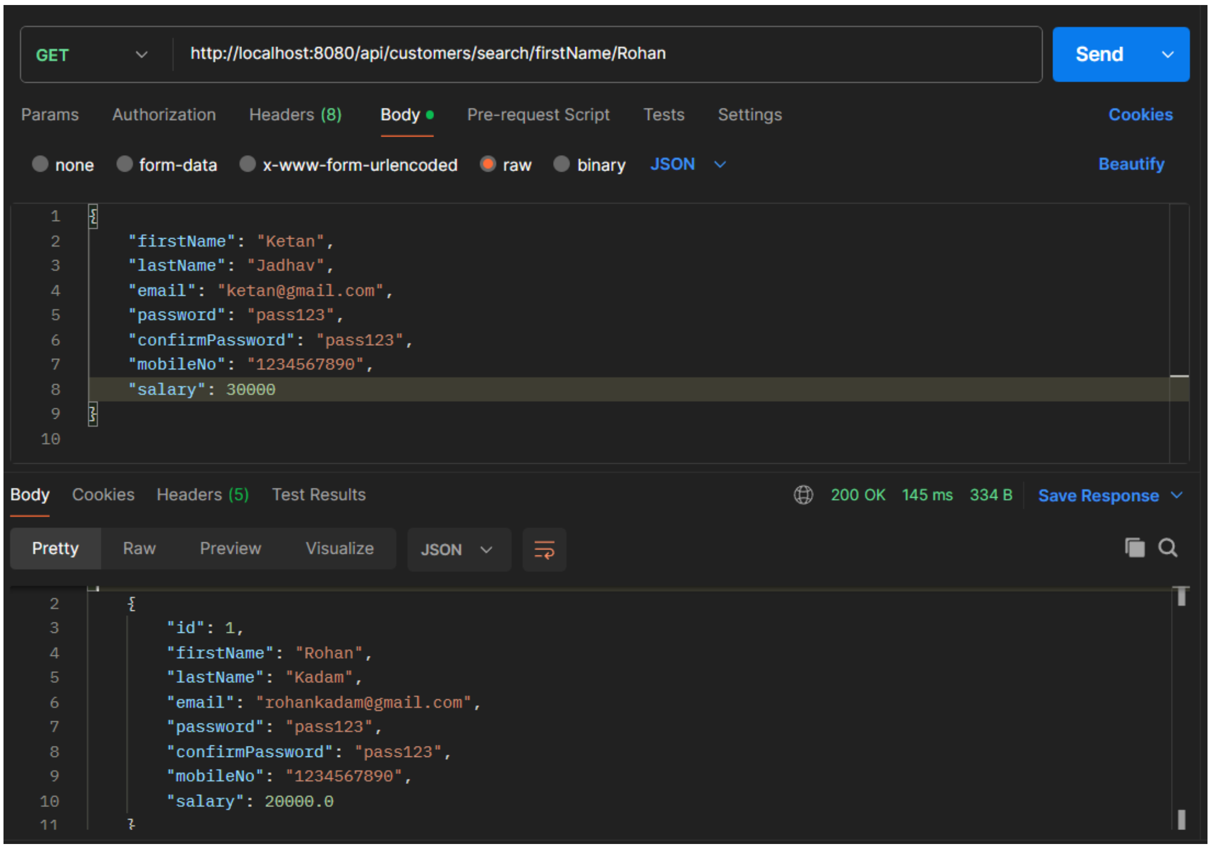
POST http://localhost:8080/api/customers

```
{
    "firstName": "Ketan",
    "lastName": "Jadhav",
    "email": "ketan@gmail.com",
    "password": "pass123",
    "confirmPassword": "pass123",
    "mobileNo": "1234567890",
    "salary": 30000
}
```

200 OK 33 ms 328 B

```
{
    "id": 2,
    "firstName": "Ketan",
    "lastName": "Jadhav",
    "email": "ketan@gmail.com",
    "password": "pass123",
    "confirmPassword": "pass123",
    "mobileNo": "1234567890",
    "salary": 30000.0
}
```

Find by First Name

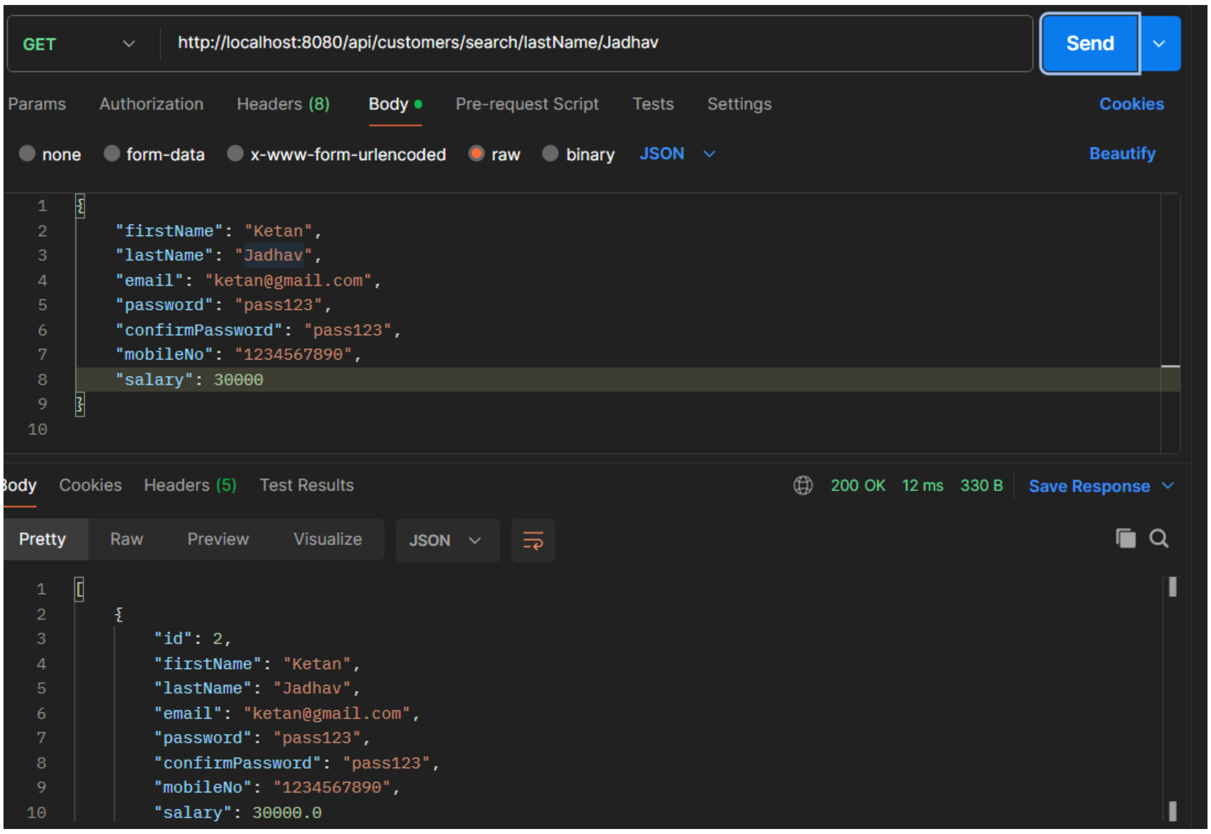

GET http://localhost:8080/api/customers/search/firstName/Rohan

```
{
    "firstName": "Ketan",
    "lastName": "Jadhav",
    "email": "ketan@gmail.com",
    "password": "pass123",
    "confirmPassword": "pass123",
    "mobileNo": "1234567890",
    "salary": 30000
}
```

200 OK 145 ms 334 B

```
    {
        "id": 1,
        "firstName": "Rohan",
        "lastName": "Kadam",
        "email": "rohankadam@gmail.com",
        "password": "pass123",
        "confirmPassword": "pass123",
        "mobileNo": "1234567890",
        "salary": 20000.0
    }
```

Find by Last Name

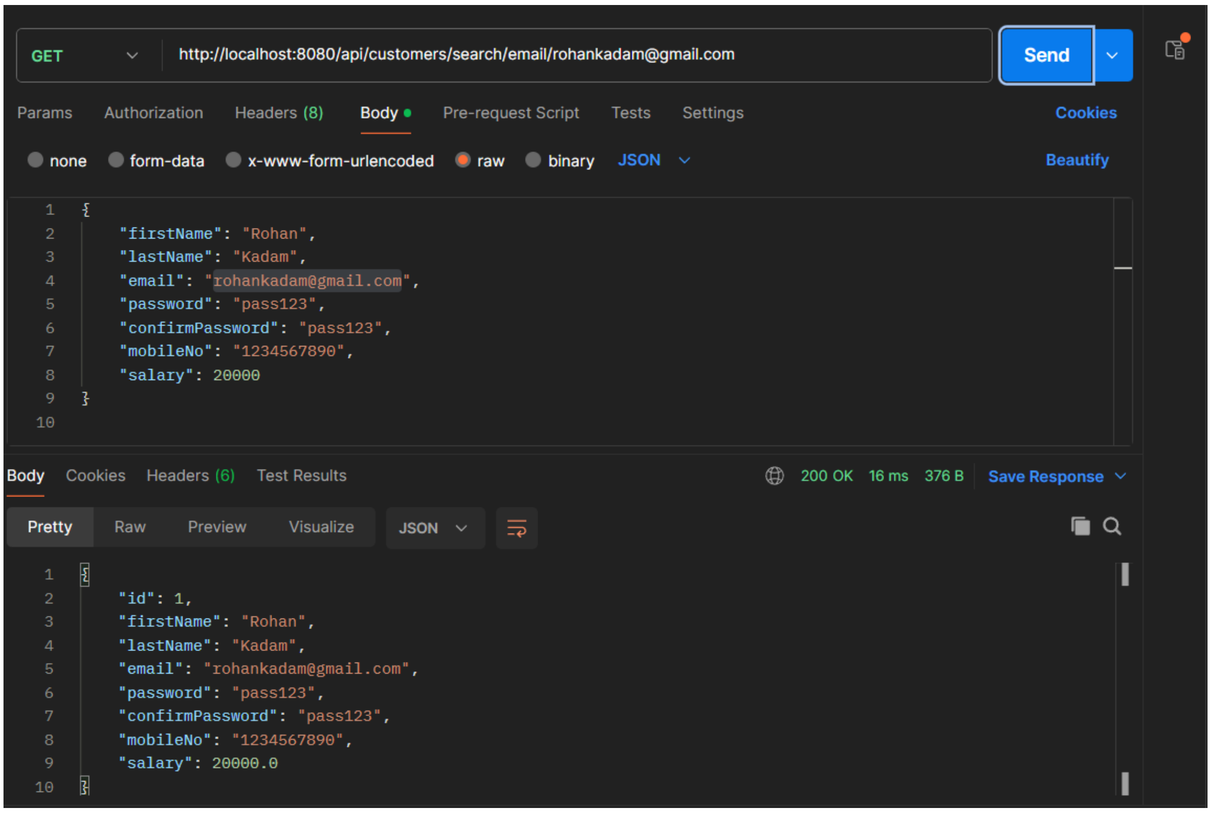

GET http://localhost:8080/api/customers/search/lastName/Jadhav

```
{
    "firstName": "Ketan",
    "lastName": "Jadhav",
    "email": "ketan@gmail.com",
    "password": "pass123",
    "confirmPassword": "pass123",
    "mobileNo": "1234567890",
    "salary": 30000
}
```

200 OK 12 ms 330 B

```
[
    {
        "id": 2,
        "firstName": "Ketan",
        "lastName": "Jadhav",
        "email": "ketan@gmail.com",
        "password": "pass123",
        "confirmPassword": "pass123",
        "mobileNo": "1234567890",
        "salary": 30000.0
```

Find by Email



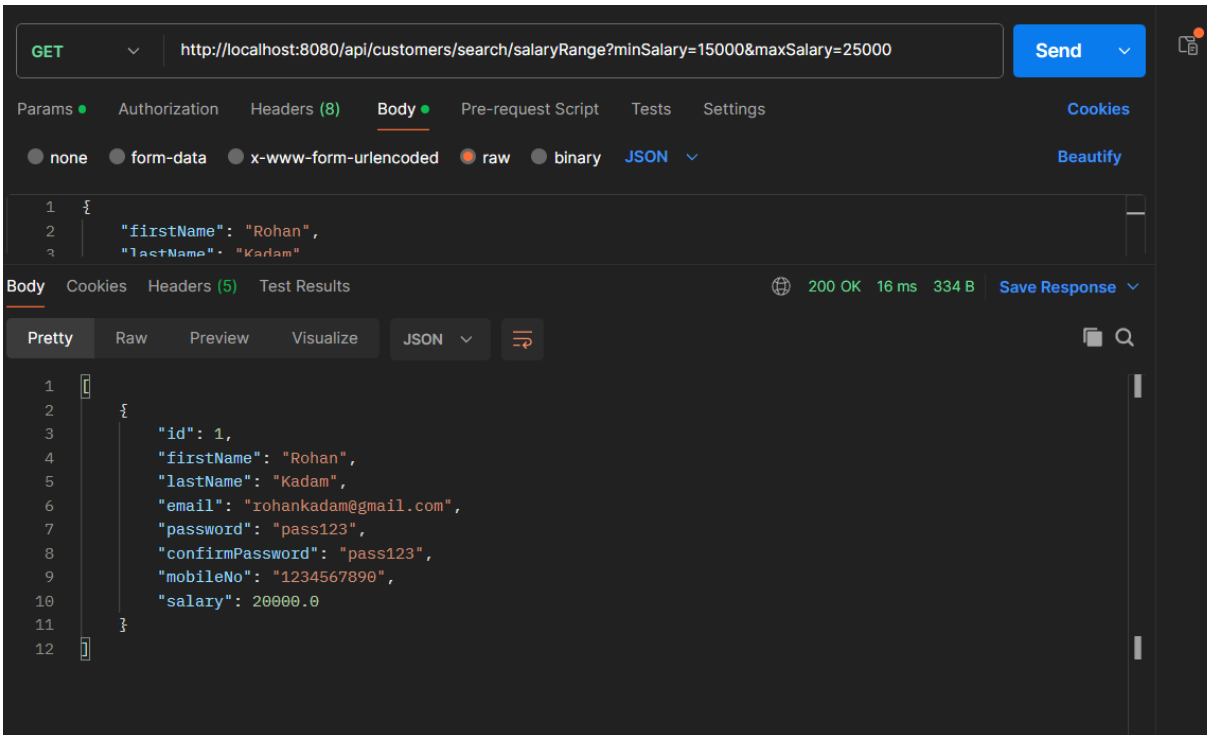
```
http://localhost:8080/api/customers/search/email/rohankadam@gmail.com
```

```
{
    "firstName": "Rohan",
    "lastName": "Kadam",
    "email": "rohankadam@gmail.com",
    "password": "pass123",
    "confirmPassword": "pass123",
    "mobileNo": "1234567890",
    "salary": 20000
}
```

200 OK 16 ms 376 B

```
{
    "id": 1,
    "firstName": "Rohan",
    "lastName": "Kadam",
    "email": "rohankadam@gmail.com",
    "password": "pass123",
    "confirmPassword": "pass123",
    "mobileNo": "1234567890",
    "salary": 20000.0
}
```

Find by Salary Range



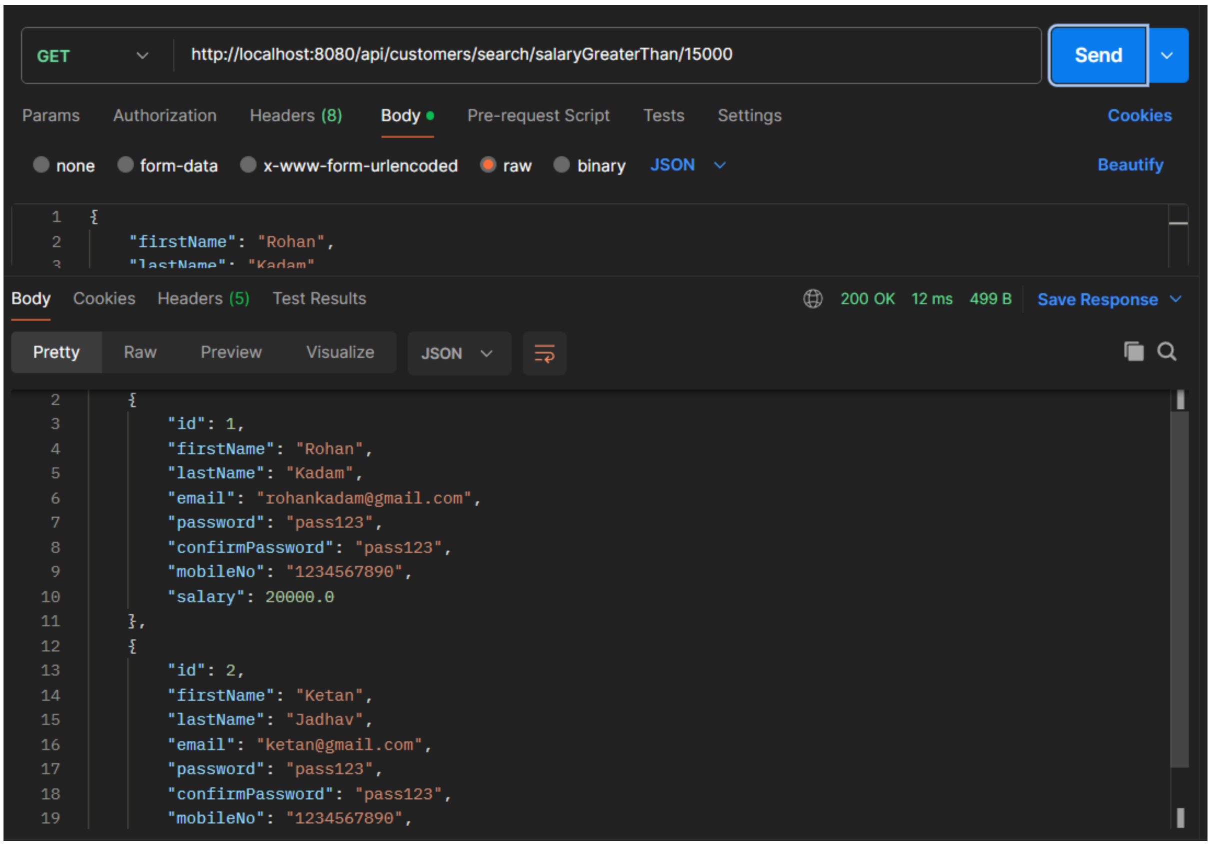
```
http://localhost:8080/api/customers/search/salaryRange?minSalary=15000&maxSalary=25000
```

```
{
    "firstName": "Rohan",
    "lastName": "Kadam"
```

200 OK 16 ms 334 B

```
[
    {
        "id": 1,
        "firstName": "Rohan",
        "lastName": "Kadam",
        "email": "rohankadam@gmail.com",
        "password": "pass123",
        "confirmPassword": "pass123",
        "mobileNo": "1234567890",
        "salary": 20000.0
    }
]
```

Find by Salary Greater Than



```
GET http://localhost:8080/api/customers/search/salaryGreaterThan/15000
```

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary JSON Beautify

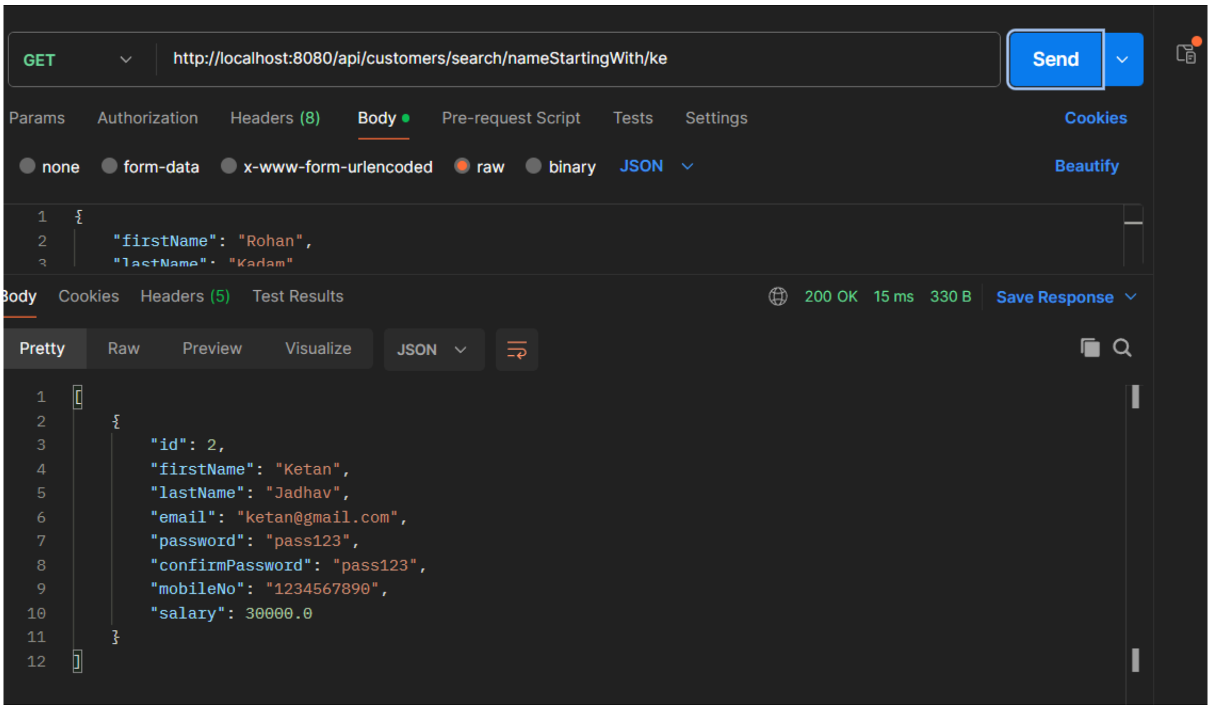
```
{
    "firstName": "Rohan",
    "lastName": "Kadam"
```

Body Cookies Headers (5) Test Results 200 OK 12 ms 499 B Save Response

Pretty Raw Preview Visualize JSON

```
    {
        "id": 1,
        "firstName": "Rohan",
        "lastName": "Kadam",
        "email": "rohankadam@gmail.com",
        "password": "pass123",
        "confirmPassword": "pass123",
        "mobileNo": "1234567890",
        "salary": 20000.0
    },
    {
        "id": 2,
        "firstName": "Ketan",
        "lastName": "Jadhav",
        "email": "ketan@gmail.com",
        "password": "pass123",
        "confirmPassword": "pass123",
        "mobileNo": "1234567890",
```

Find by Name Starting With



```
GET http://localhost:8080/api/customers/search/nameStartingWith/ke
```

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary JSON Beautify

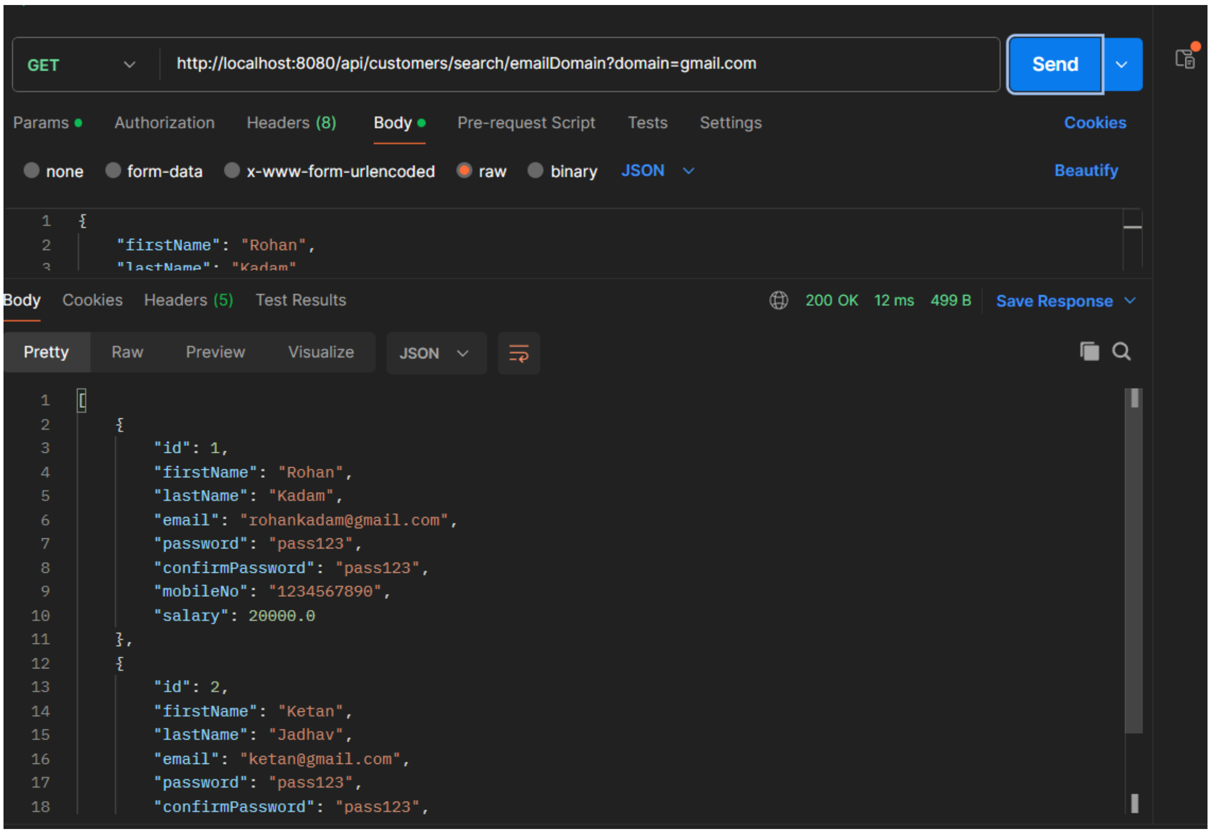
```
{
    "firstName": "Rohan",
    "lastName": "Kadam"
```

Body Cookies Headers (5) Test Results 200 OK 15 ms 330 B Save Response

Pretty Raw Preview Visualize JSON

```
[
    {
        "id": 2,
        "firstName": "Ketan",
        "lastName": "Jadhav",
        "email": "ketan@gmail.com",
        "password": "pass123",
        "confirmPassword": "pass123",
        "mobileNo": "1234567890",
        "salary": 30000.0
    }
]
```

Find by Email Domain



http://localhost:8080/api/customers/search/emailDomain?domain=gmail.com

```
{
    "firstName": "Rohan",
    "lastName": "Kadam"
```

```
[
    {
        "id": 1,
        "firstName": "Rohan",
        "lastName": "Kadam",
        "email": "rohankadam@gmail.com",
        "password": "pass123",
        "confirmPassword": "pass123",
        "mobileNo": "1234567890",
        "salary": 20000.0
    },
    {
        "id": 2,
        "firstName": "Ketan",
        "lastName": "Jadhav",
        "email": "ketan@gmail.com",
        "password": "pass123",
        "confirmPassword": "pass123",
```