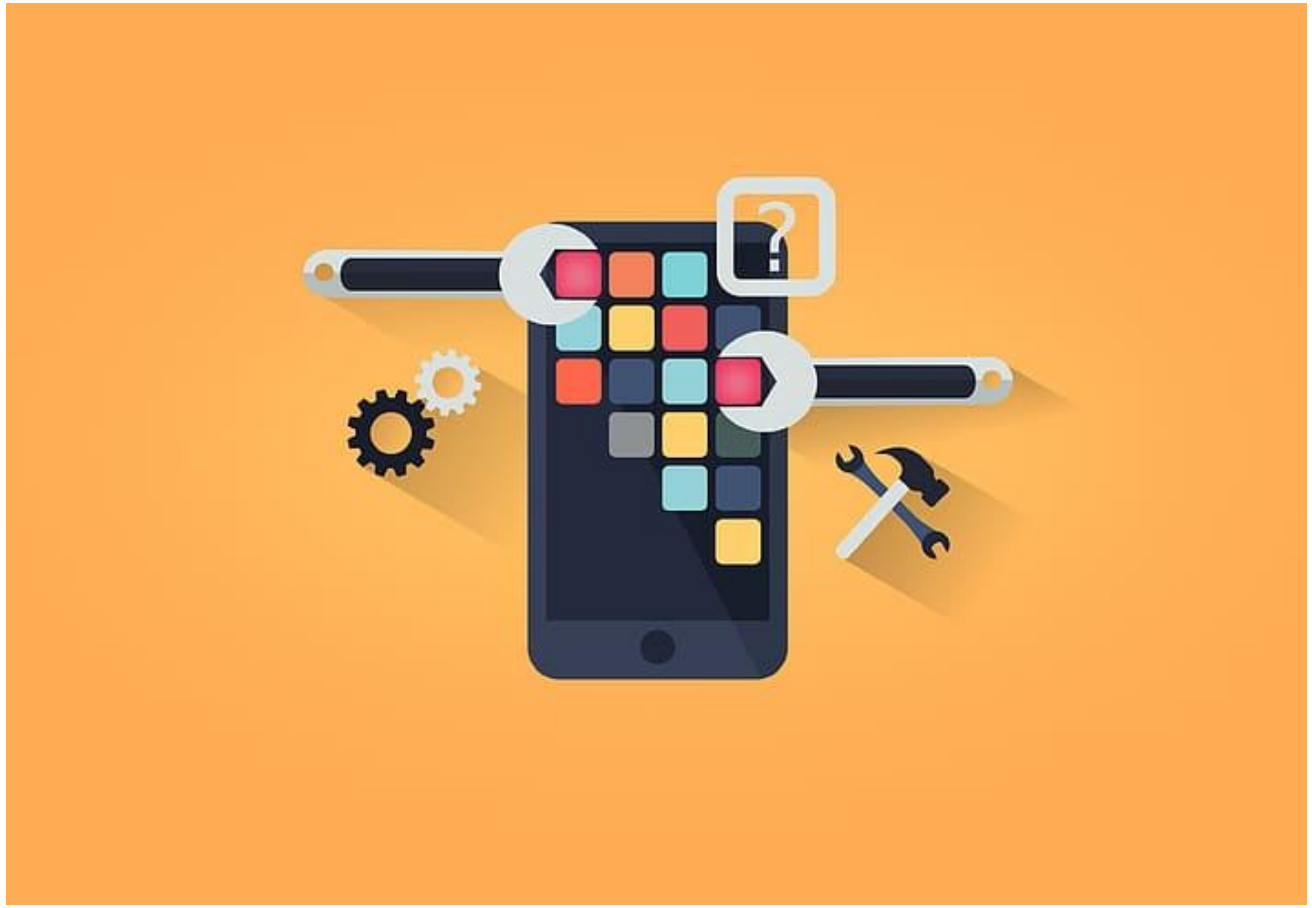
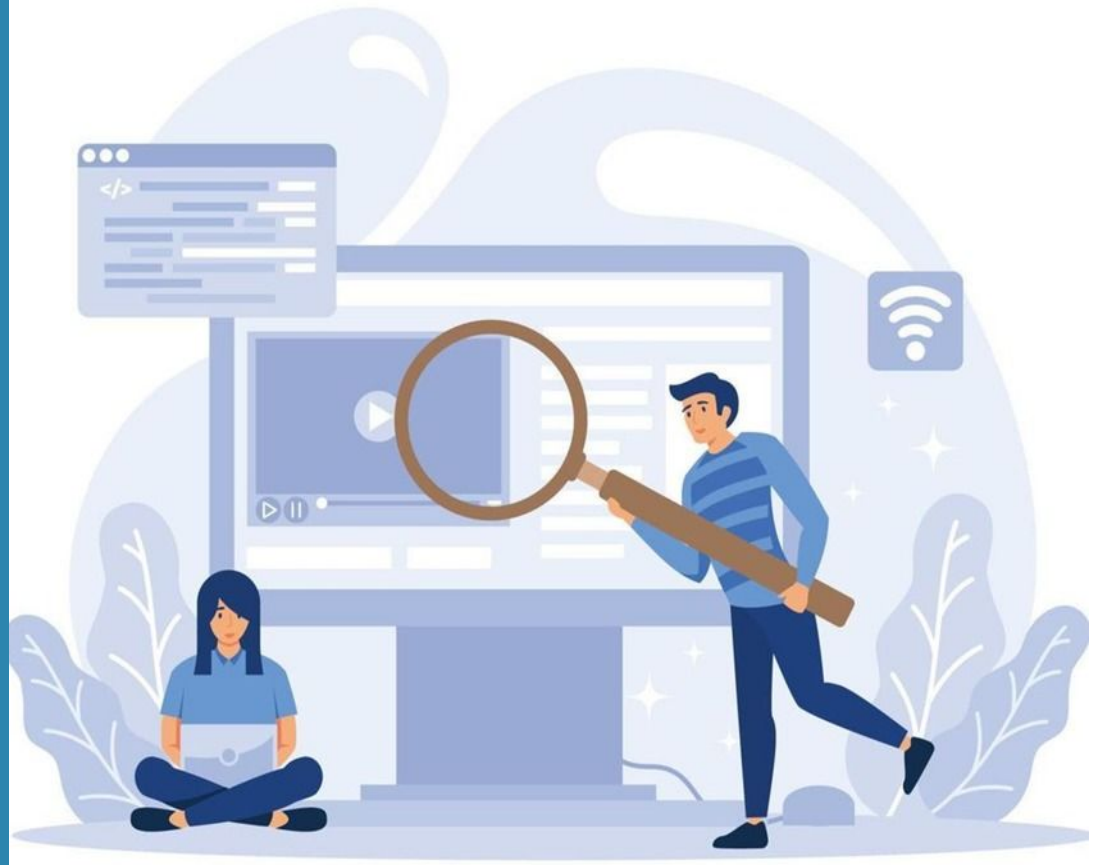


E-Medicare

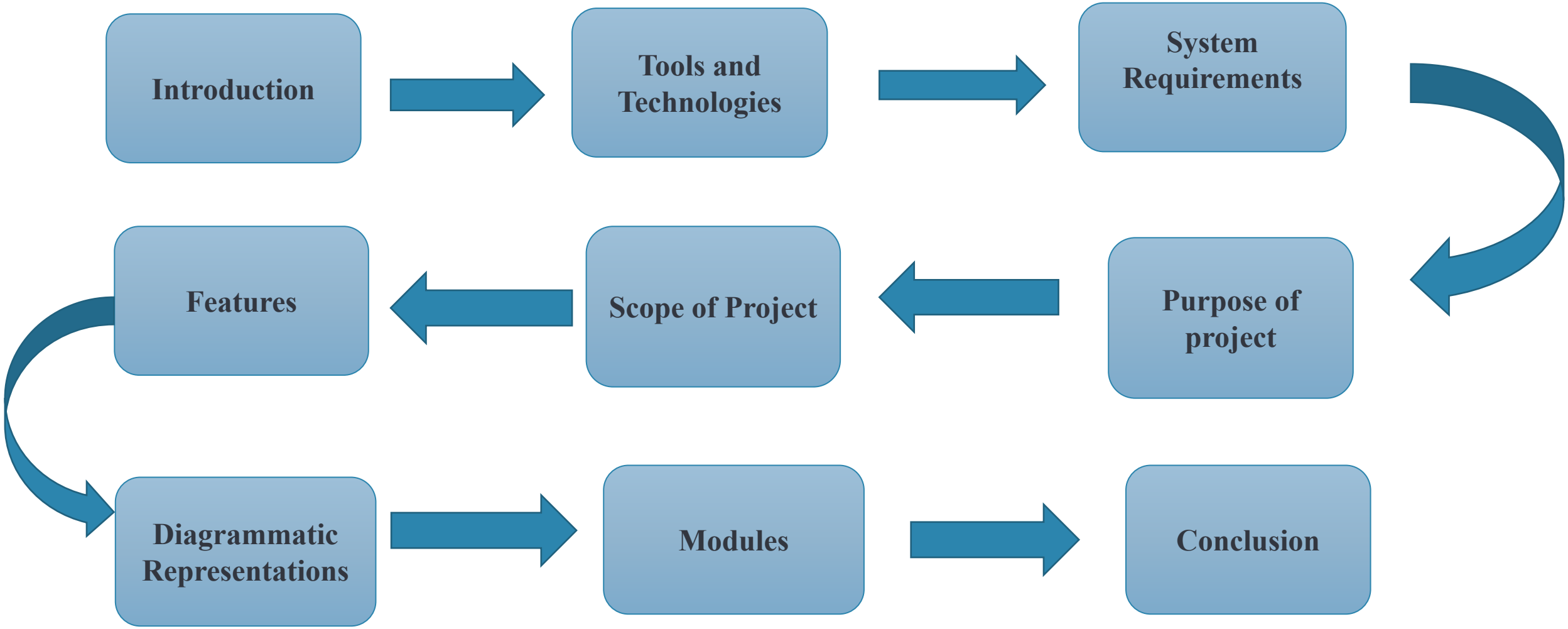
-GROUP 5



Group-5

- Harsh Rathod -2563088
- Swaroop A V -2562956
- Sulagna Chatterjee -2563990
- Priya Singha -2564017
- Sinchana Rao -2564007
- Pavithra S -2564183
- Rajalakshmi H -2564069
- Vallepu Sree Lakshmi Kavya -2563318
- Varshita D C -2563588
- Yendru Pavan Kumar -2564326

Contents


Introduction
Tools and Technologies
System Requirements
Purpose of project
Scope of Project
Features
Diagrammatic Representations
Modules
Conclusion

Tools And Technologies

Back End

- MySQL
- Spring Boot
- Postman
- Rapid API
- Docker
- RabbitMq

Cloud

- Cloud Config
- Netflix Eureka
- Open Feign
- Resilience4j
- Retry(Circuit Breaker)
- Zipkin

System Requirements

The Basic System Requirements for Running this project are listed below:

- ❑ Visual studio/Eclipse, Postman , Docker and MySQL are to be installed to the system
- ❑ 1 GB Random Access Memory
- ❑ 200 MB of Free Space on Hard Disk
- ❑ Microsoft Windows 8,10,11 or Linux or Equivalent OS
- ❑ Web Browser (Microsoft Edge, Google Chrome, Firefox)

Introduction

INTRODUCTION

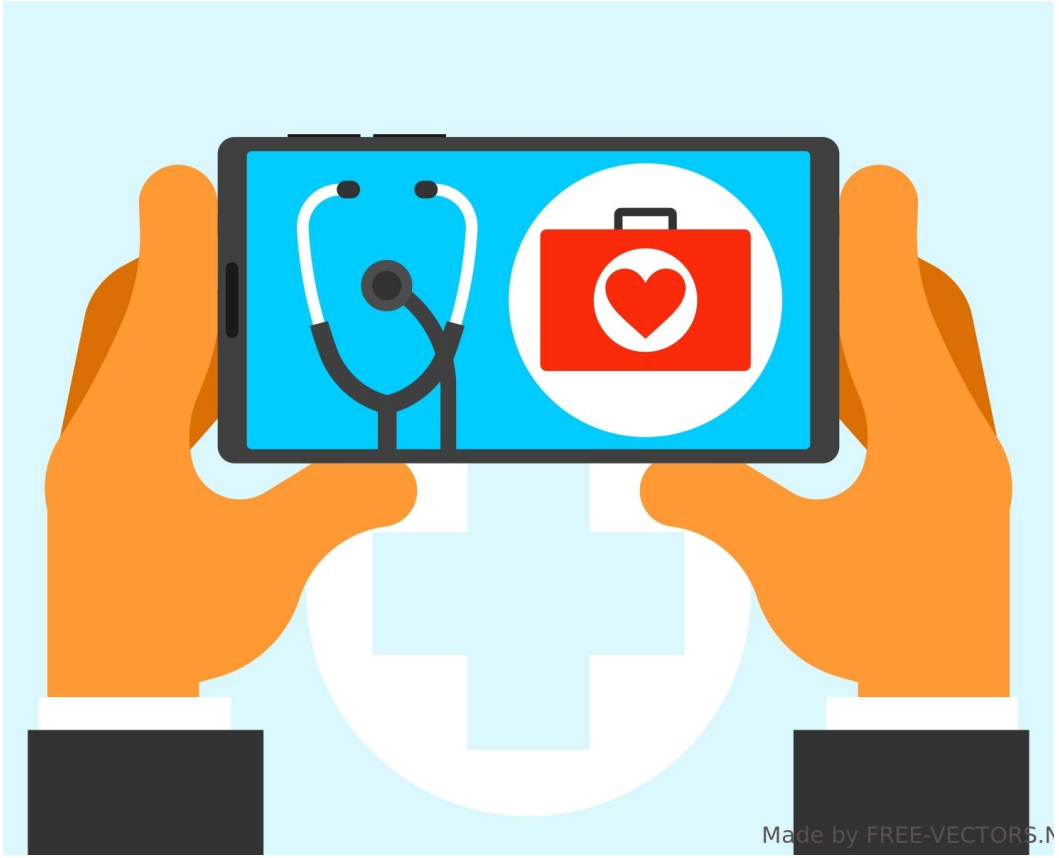
- E-Medicare is a modular and distributed system built upon the principles of microservices architecture, a software development approach that breaks down complex applications into smaller, interconnected components or services.
- Each microservice is designed to perform a specific function, such as register service, login service,cart service, payment service and more.
- The key advantages of Scalability, Interoperability, Data Security,Telemedicine Integration, Cost-Efficiency and more.

PURPOSE:

- E-Medicare microservices break down complex healthcare systems into smaller, specialized components.
- They facilitate seamless integration with existing healthcare systems, ensuring that E-Medicare services can work together with other healthcare applications, databases, and technologies.
- By optimizing resource allocation and reducing operational overhead, E-Medicare microservices can help healthcare organizations reduce costs while maintaining or even improving the quality of care delivered to Medicare beneficiaries.
- E-Medicare microservices generate valuable healthcare data that can be leveraged for data analytics and insights.

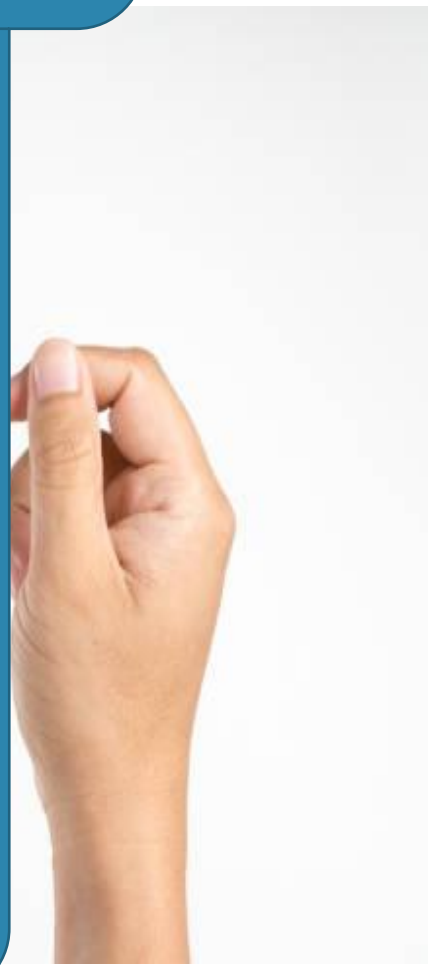
Scope

- Online Healthcare product delivery
- Telemedicine Integration
- Billing and Claims Processing
- User Management
- Scalability and Flexibility



SERVICES

- Eureka Server
- Config Server
- Register Service
- Login Service
- Dashboard Service
- Product Service
- Management Service
- Change password Service
- Cart Service
- Address Service
- Payment Service
- Shipping Server



Features Available

Admin

- Register
- Login
- Manage Product
- Manage User Login creds

User

- Register
- Login
- Search,view, purchaseProduct
- Add Address

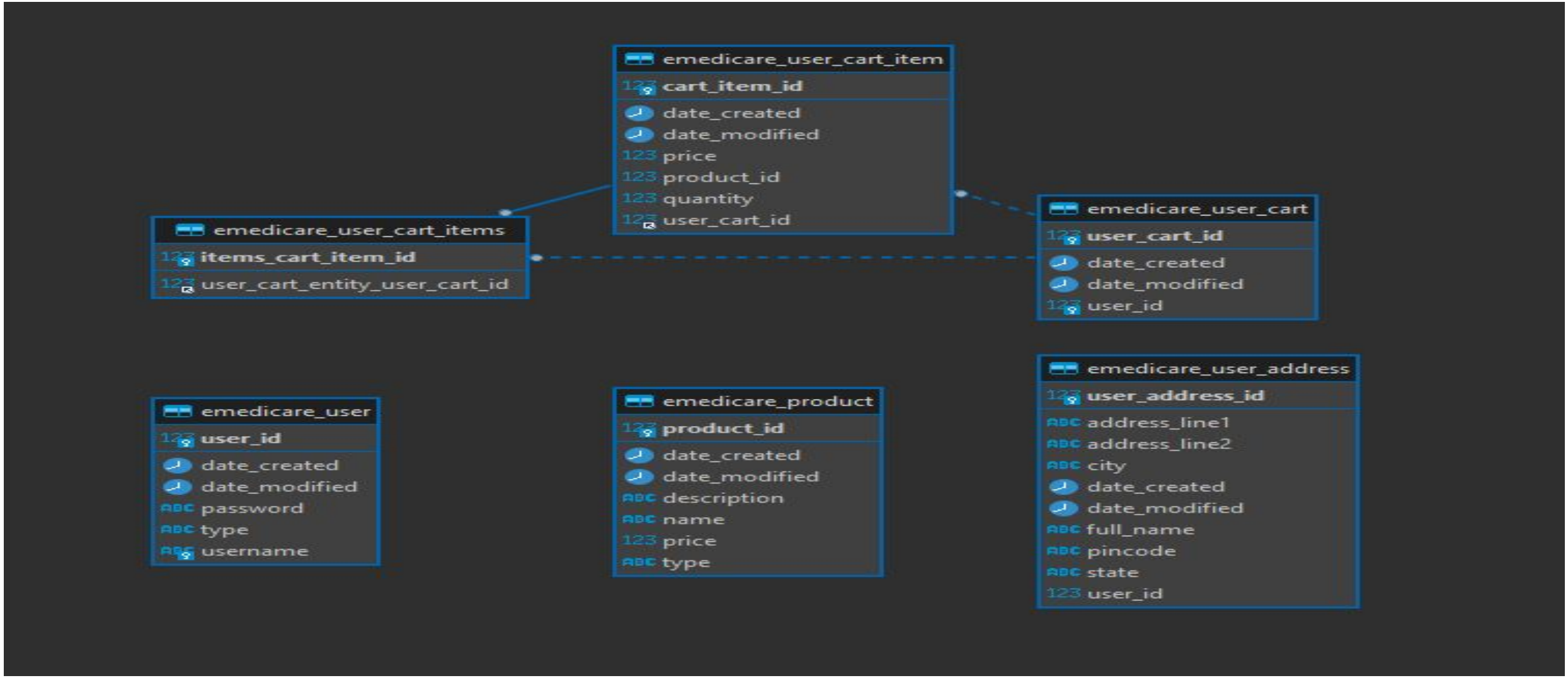
Product

- Add Product
- Update Product
- Delete Product

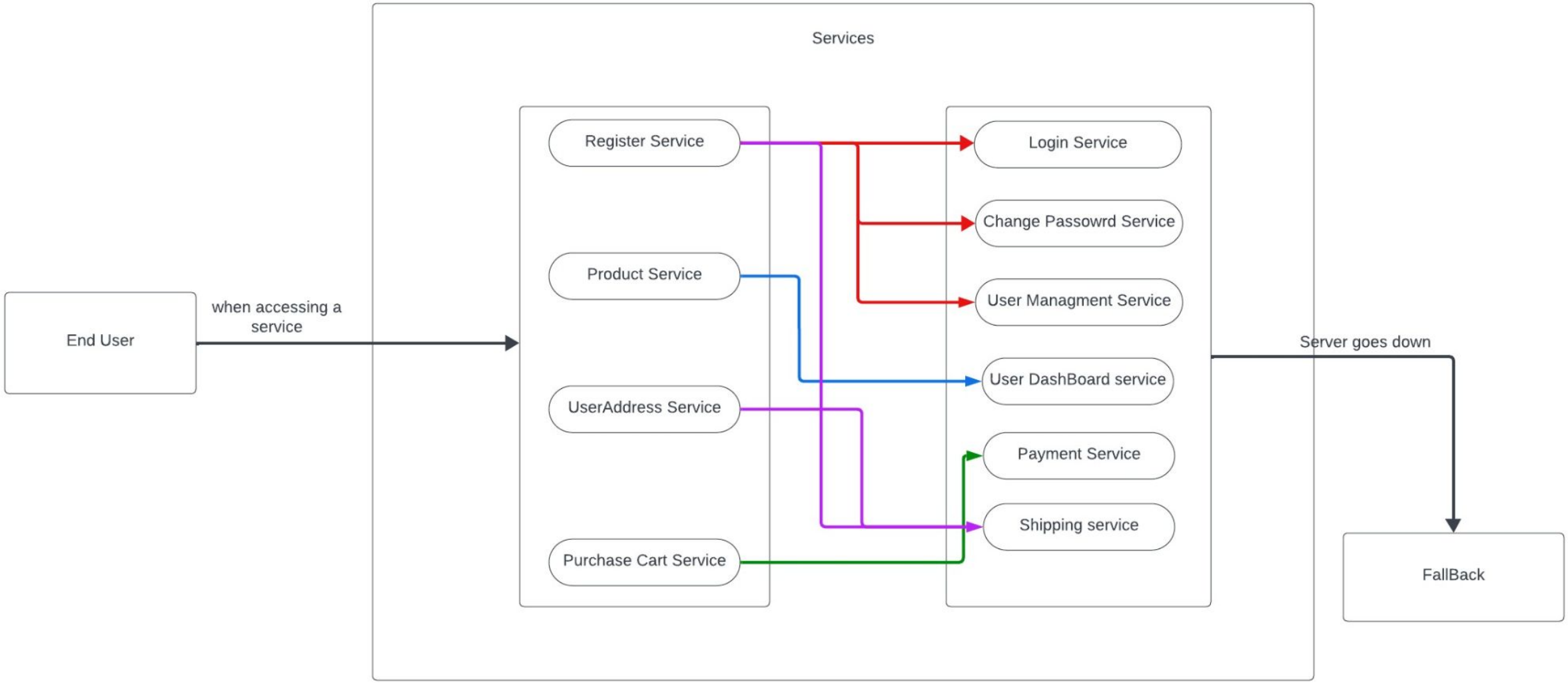
Payment

- Place Order
- Cash on Delivery
- Add Card
- Delete Card
- Payment

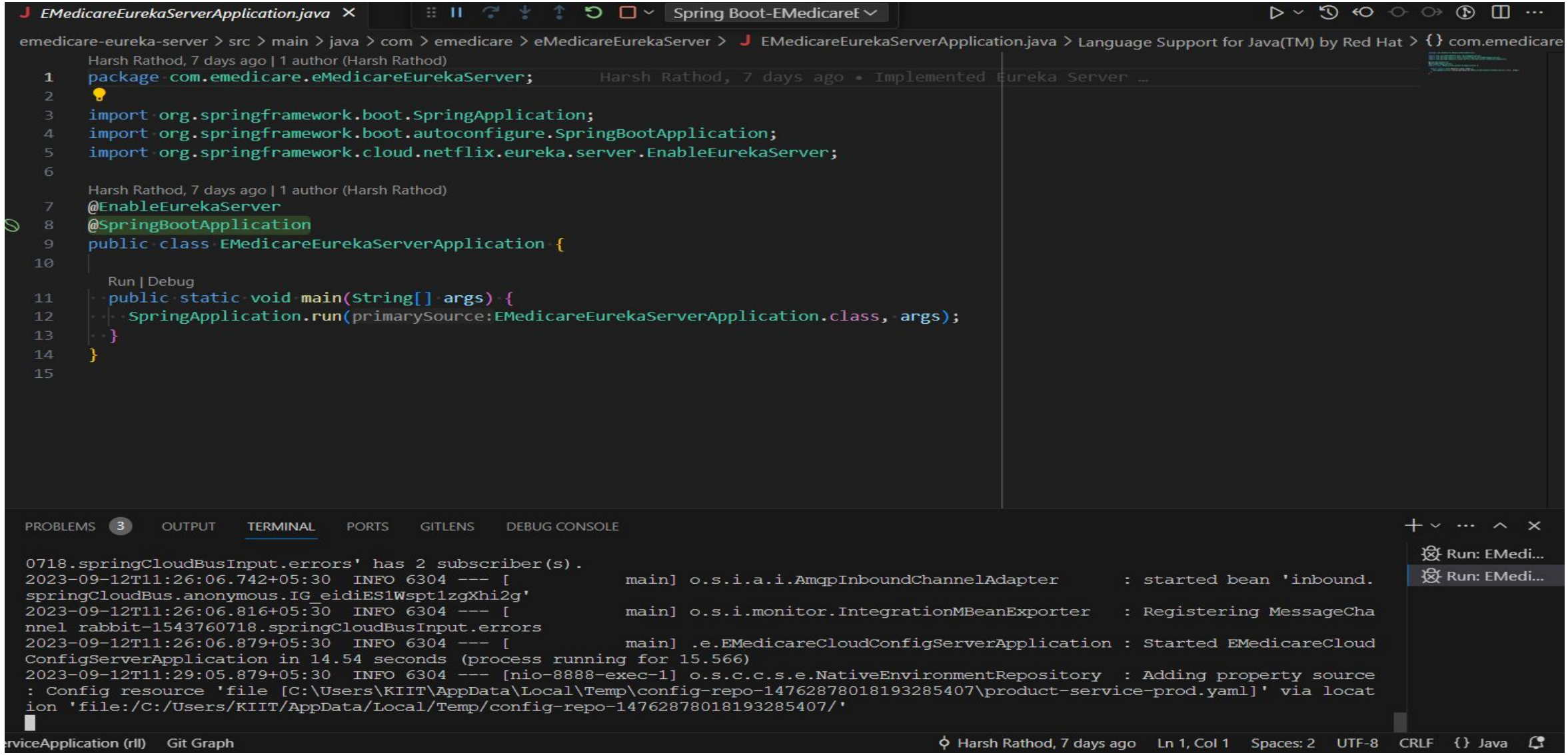
Database Schema



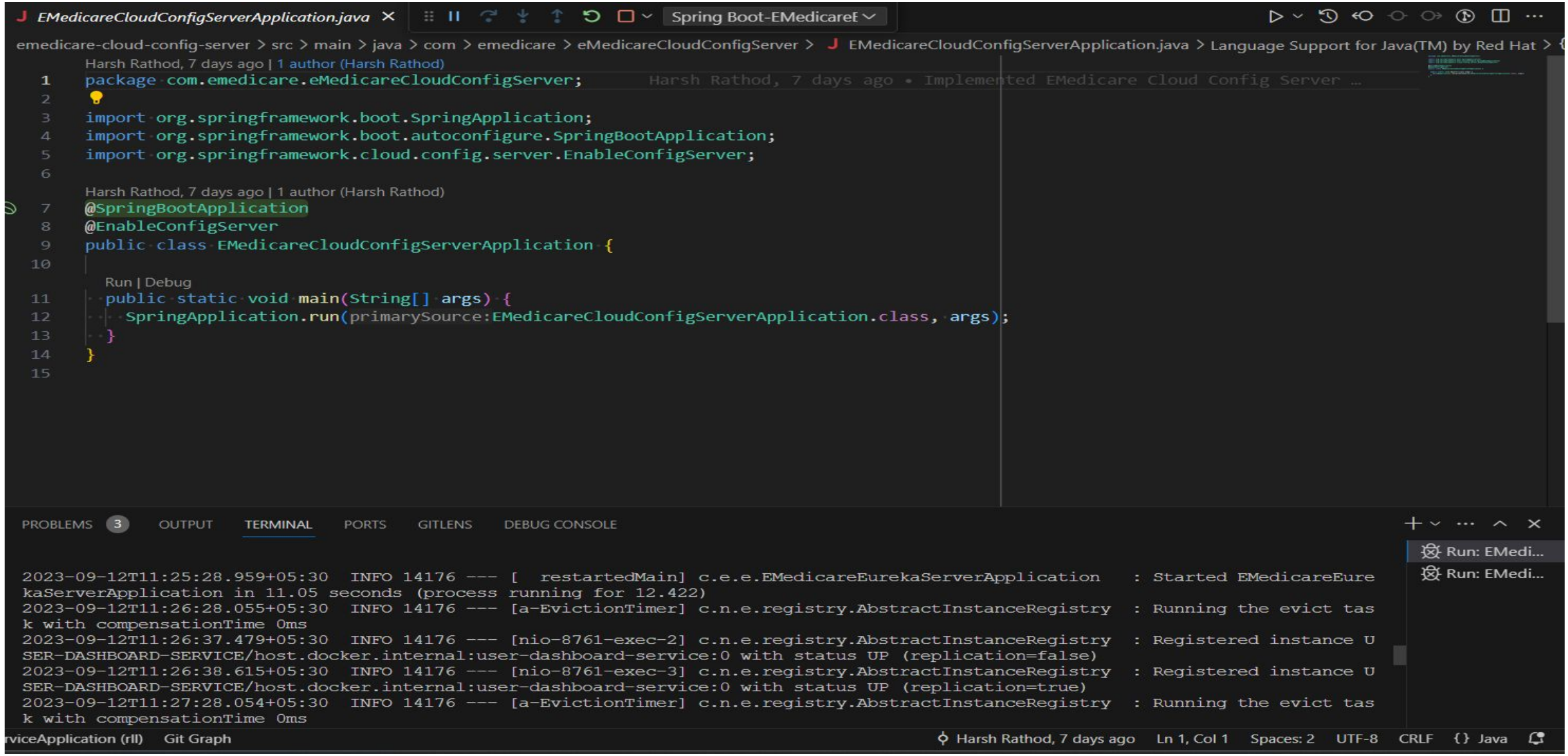
FlowChart


Services
Register Service
Product Service
UserAddress Service
Purchase Cart Service
Login Service
Change Passowrd Service
User Managment Service
User DashBoard service
Payment Service
Shipping service
End User
when accessing a service
Server goes down
FallBack

Eureka Server



Config Server



Register

sulagna

Logic

Main part:

1.Create microService

2.Connect with Config and Eureka Server

Secondary part:

1. Feign Client
2. Reselience4j
3. Retry

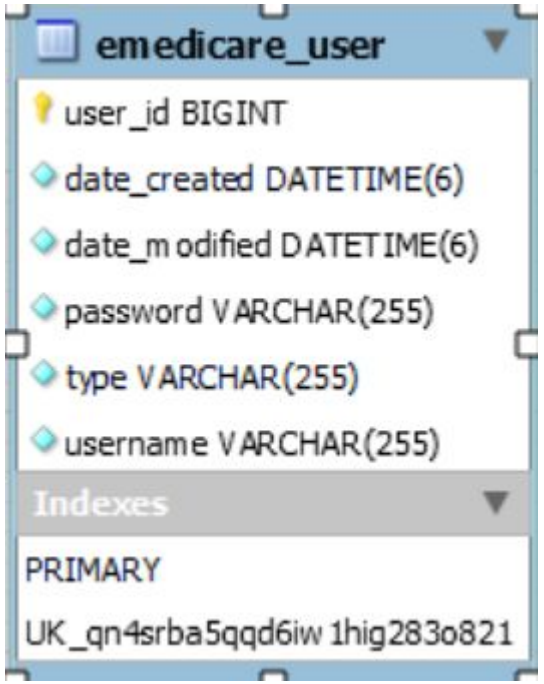
Table

Table: emedicare_user

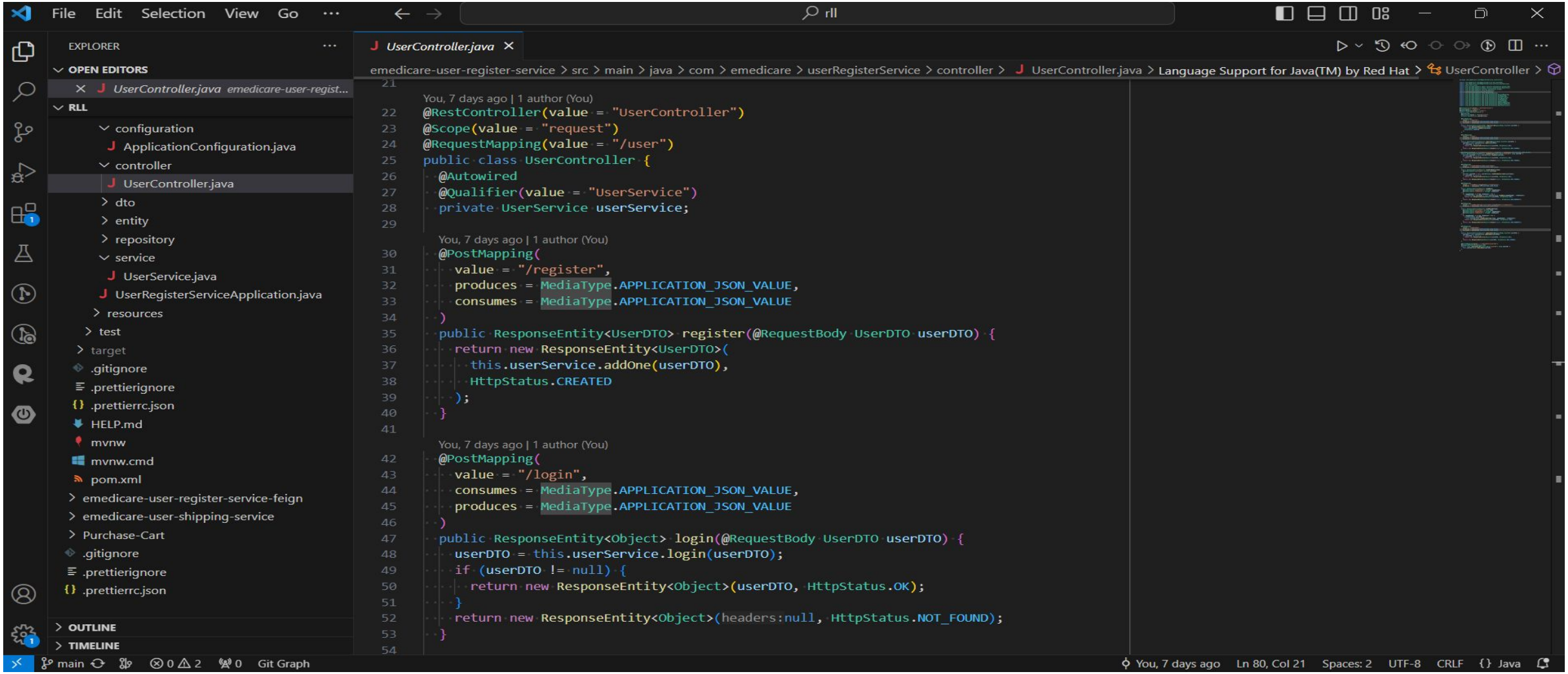
Columns:

user_id	bigint AI PK
date_created	datetime(6)
date_modified	datetime(6)
password	varchar(255)
type	varchar(255)
username	varchar(255)

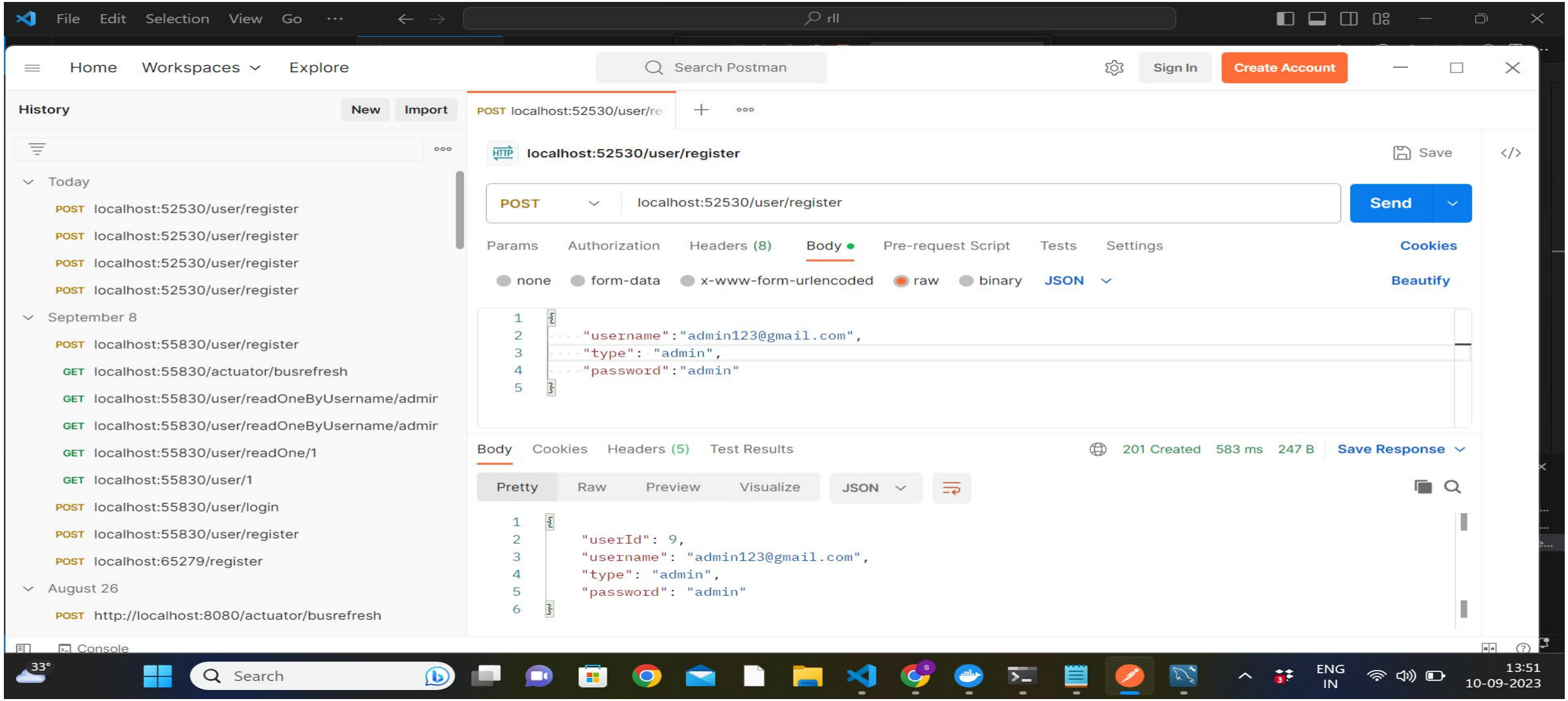
ER Diagram



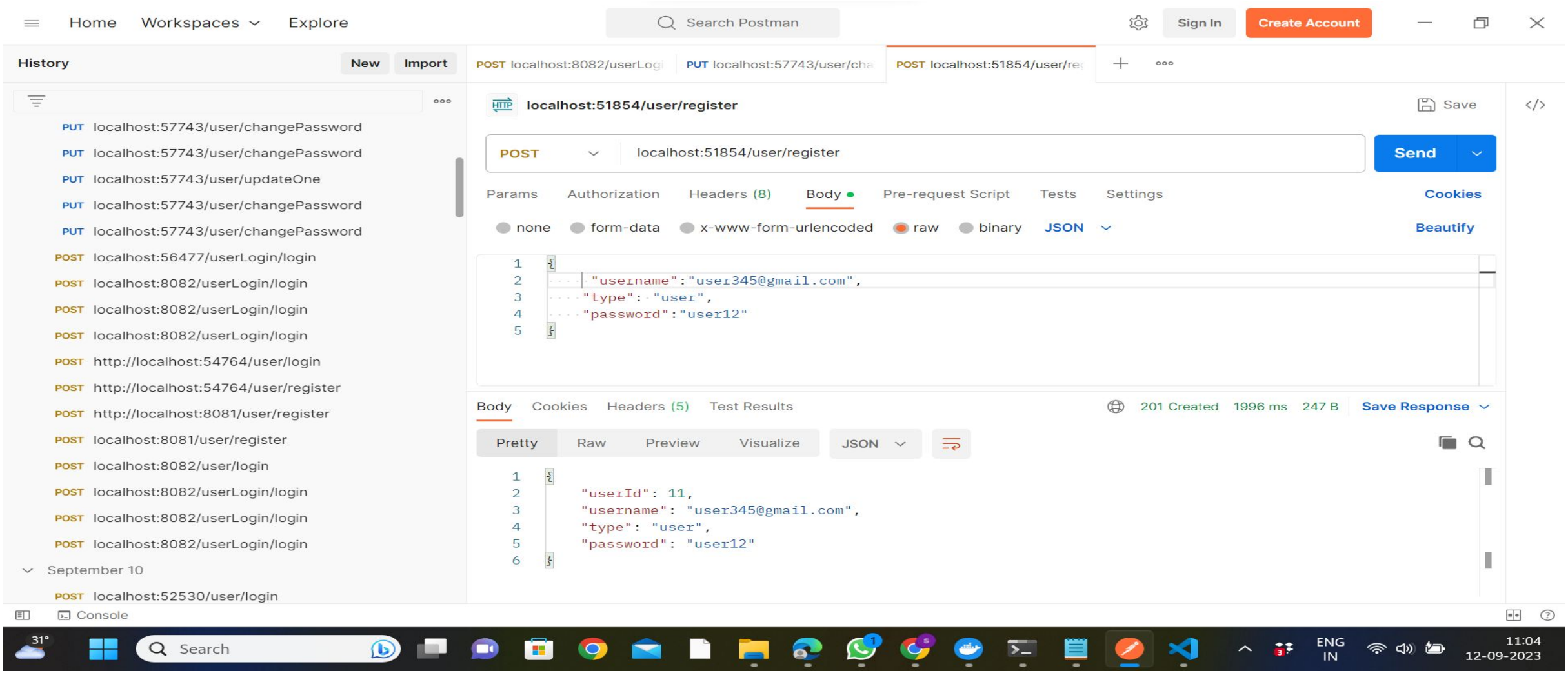
MicroService



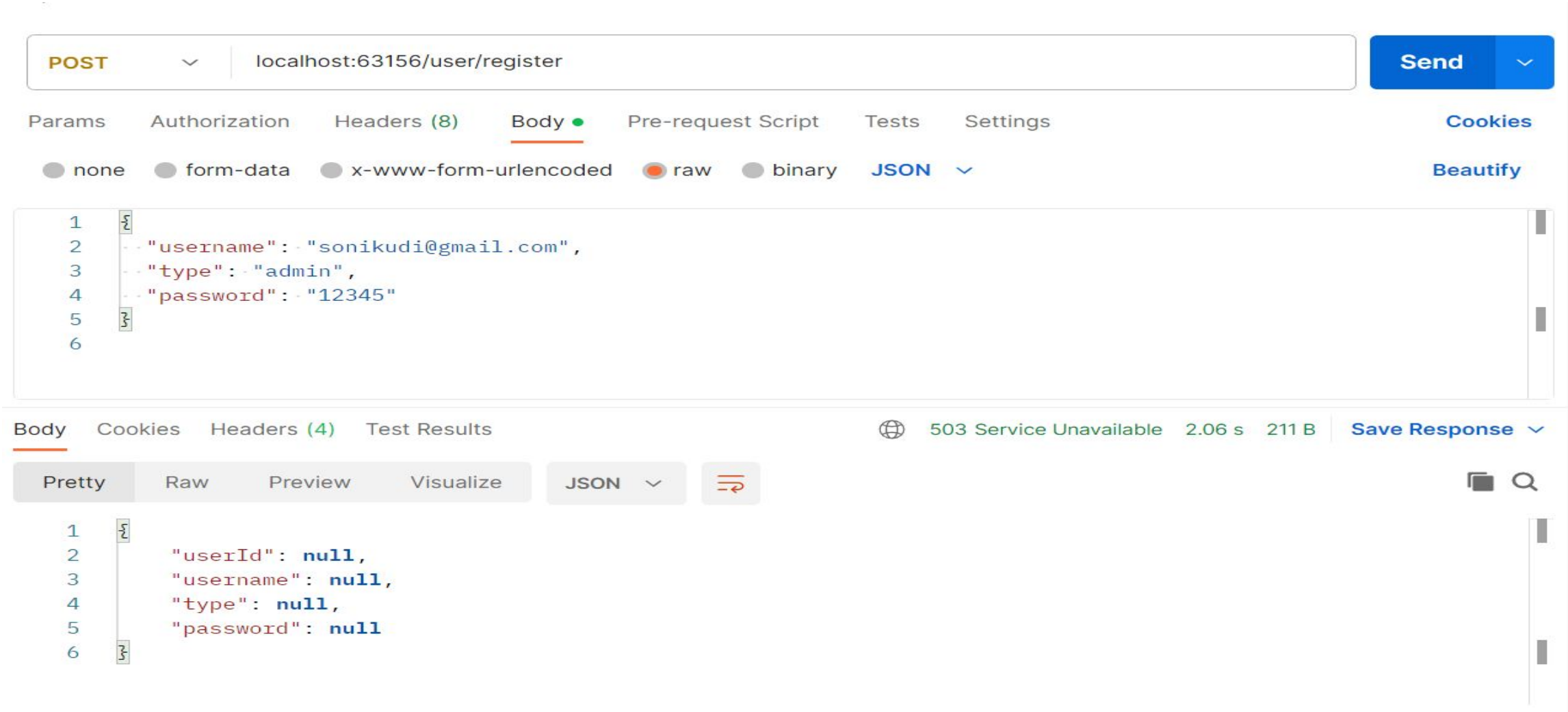
Register as Admin



Register as User



Fallback


POST
localhost:63156/user/register
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
JSON
Beautify
{
"username": "sonikudi@gmail.com",
"type": "admin",
"password": "12345"
}
Body
Cookies
Headers (4)
Test Results
503 Service Unavailable
2.06 s
211 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"userId": null,
"username": null,
"type": null,
"password": null
}

Git

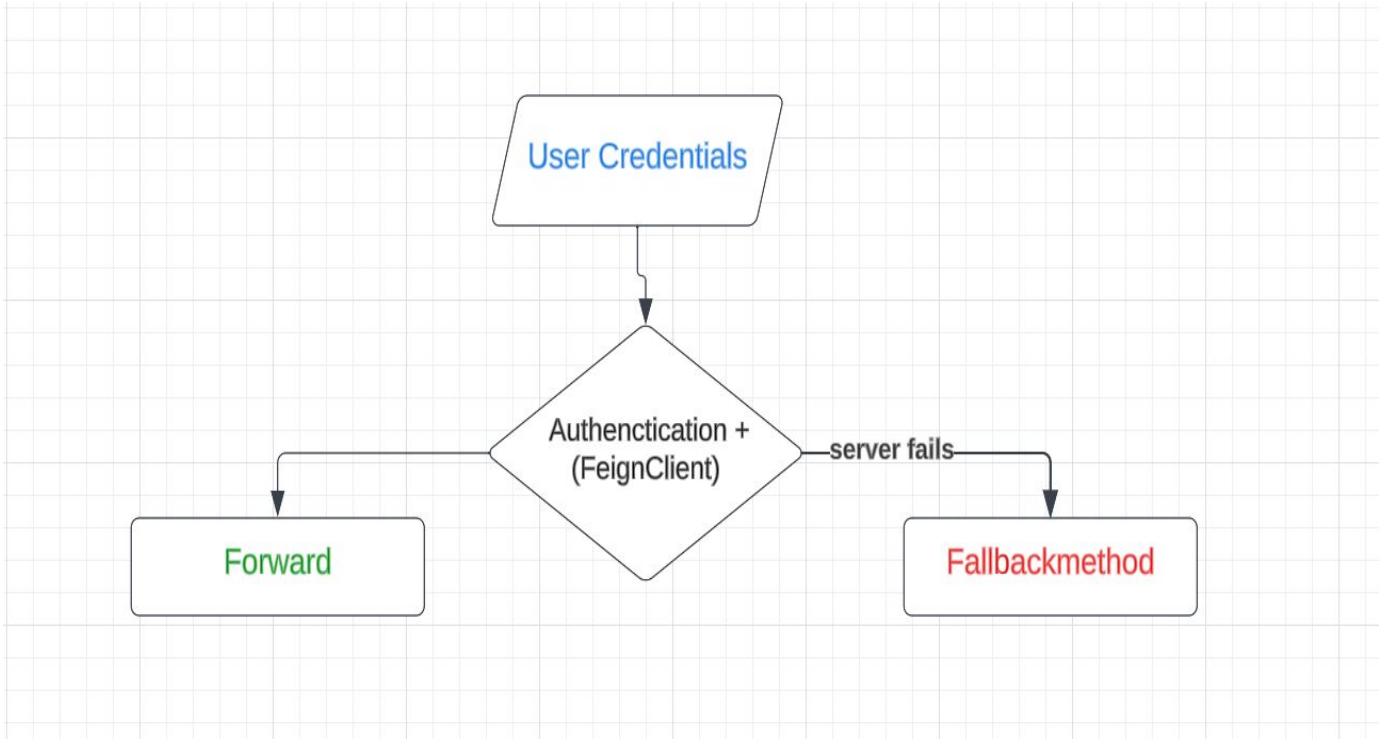
```
commit d863cace9f0994a297b6b21c745a41d39e6864f6 (HEAD -> main, origin/main, origin/HEAD)
Author: sulagna <sulagnachatterjee27@gmail.com>
Date:   Tue Sep 12 16:15:12 2023 +0530

    Register Feign

commit 828f8b39c37b6f505b87ecf9c39db6fa99b9a7c8
Author: sulagna <sulagnachatterjee27@gmail.com>
Date:   Wed Sep 6 14:14:01 2023 +0530

:...skipping...
commit d863cace9f0994a297b6b21c745a41d39e6864f6 (HEAD -> main, origin/main, origin/HEAD)
Author: sulagna <sulagnachatterjee27@gmail.com>
Date:   Tue Sep 12 16:15:12 2023 +0530

    Register Feign

commit 828f8b39c37b6f505b87ecf9c39db6fa99b9a7c8
Author: sulagna <sulagnachatterjee27@gmail.com>
Date:   Wed Sep 6 14:14:01 2023 +0530

    Implemented user-register-service

commit bc6927a4136ce98d431710aca77d5fa26bdbe2c3
Author: sulagna <sulagnachatterjee27@gmail.com>
Date:   Sat Aug 12 10:46:16 2023 +0530

    update test

commit f6fb64f6e5d00b414a6dfae1da33f0d13652162c
Author: sulagna <sulagnachatterjee27@gmail.com>
Date:   Thu Aug 10 15:06:37 2023 +0530

    adminregistration testing

commit 3c9631cb0e4c76e521166161d3864a4497e30307
Author: sulagna <sulagnachatterjee27@gmail.com>
Date:   Thu Aug 10 12:25:12 2023 +0530

    new update

commit 56f803a086e3c5210d61d9fd131c8c6f8f2b9f0f
Author: sulagna <sulagnachatterjee27@gmail.com>
```

Login

Swaroop

- Logic

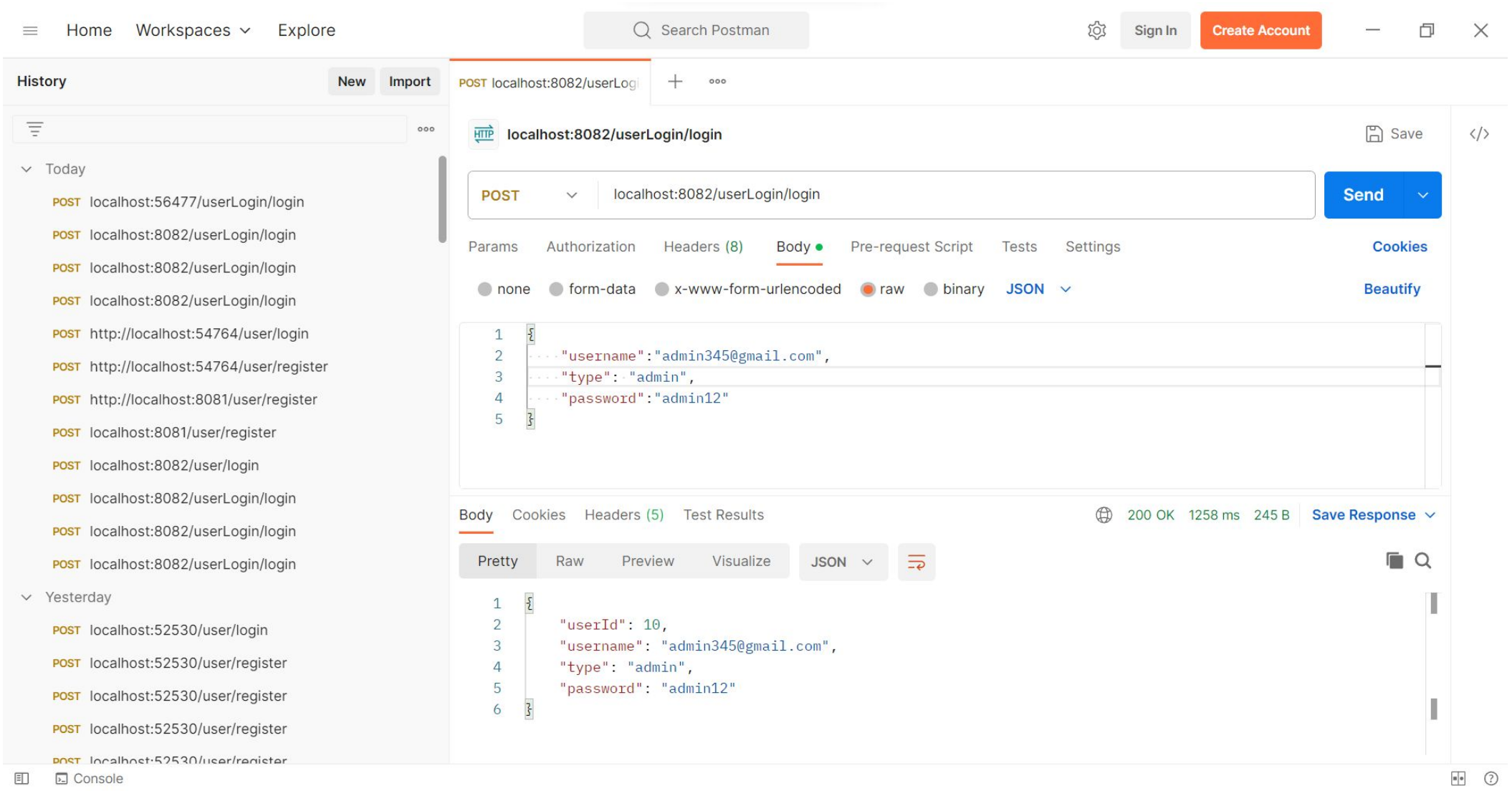
  -Feign Client
  -Resilience4j
  -Retry



Proxy Service

```
12
    You, last week | 1 author (You)
13  @FeignClient(value = "user-register-service")
14  public interface UserServiceProxy {
        You, last week | 1 author (You)
15      @PostMapping(
16          value = "/user/login",
17          consumes = MediaType.APPLICATION_JSON_VALUE,
18          produces = MediaType.APPLICATION_JSON_VALUE
19      )
20      @CircuitBreaker(name = "login", fallbackMethod = "fallbackLogin")
21      @Retry(name = "user-register-service")
22      public ResponseEntity<Object> login(@RequestBody UserDTO userDTO);
23
24      public default ResponseEntity<Object> fallbackLogin(Long userId, Exception e) {
25          try {
26              return new ResponseEntity<Object>(
27                  new UserDTO(userId:null, username:null, type:null, password:null),
28                  HttpStatus.OK
29              );
30          } catch (Exception exception) {
31              return new ResponseEntity<Object>(
32                  new UserDTO(userId:null, username:null, type:null, password:null),      You, last week • Implemente
33                  HttpStatus.NOT_FOUND
34              );
35          }
36      }
```

Login


Home
Workspaces
Explore
Search Postman
Sign In
Create Account
History
New
Import
Today
POST localhost:56477/userLogin/login
POST localhost:8082/userLogin/login
POST localhost:8082/userLogin/login
POST localhost:8082/userLogin/login
POST http://localhost:54764/user/login
POST http://localhost:54764/user/register
POST http://localhost:8081/user/register
POST localhost:8081/user/register
POST localhost:8082/user/login
POST localhost:8082/userLogin/login
POST localhost:8082/userLogin/login
POST localhost:8082/userLogin/login
Yesterday
POST localhost:52530/user/login
POST localhost:52530/user/register
POST localhost:52530/user/register
POST localhost:52530/user/register
POST localhost:8082/userLogin/login
localhost:8082/userLogin/login
Save
POST
localhost:8082/userLogin/login
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
JSON
Beautify
{
"username":"admin345@gmail.com",
"type": "admin",
"password":"admin12"
}
Body
Cookies
Headers (5)
Test Results
200 OK
1258 ms
245 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"userId": 10,
"username": "admin345@gmail.com",
"type": "admin",
"password": "admin12"
}
Console

Product

Kavya

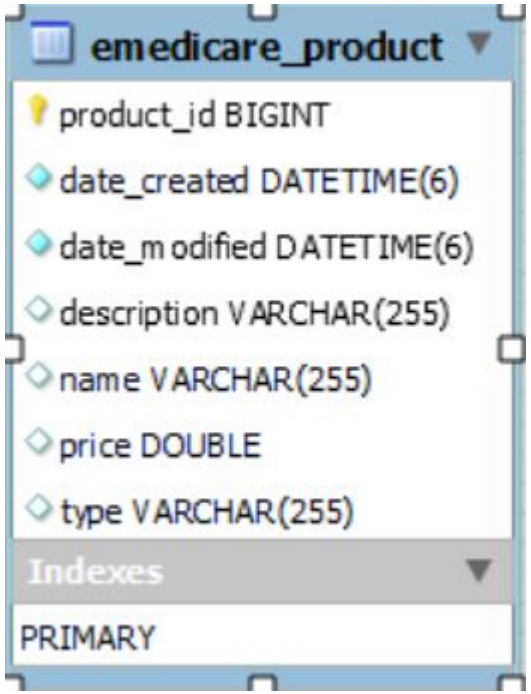
Table

Table: emedicare_product

Columns:

product_id	bigint AI PK
date_created	datetime(6)
date_modified	datetime(6)
description	varchar(255)
name	varchar(255)
price	double
type	varchar(255)

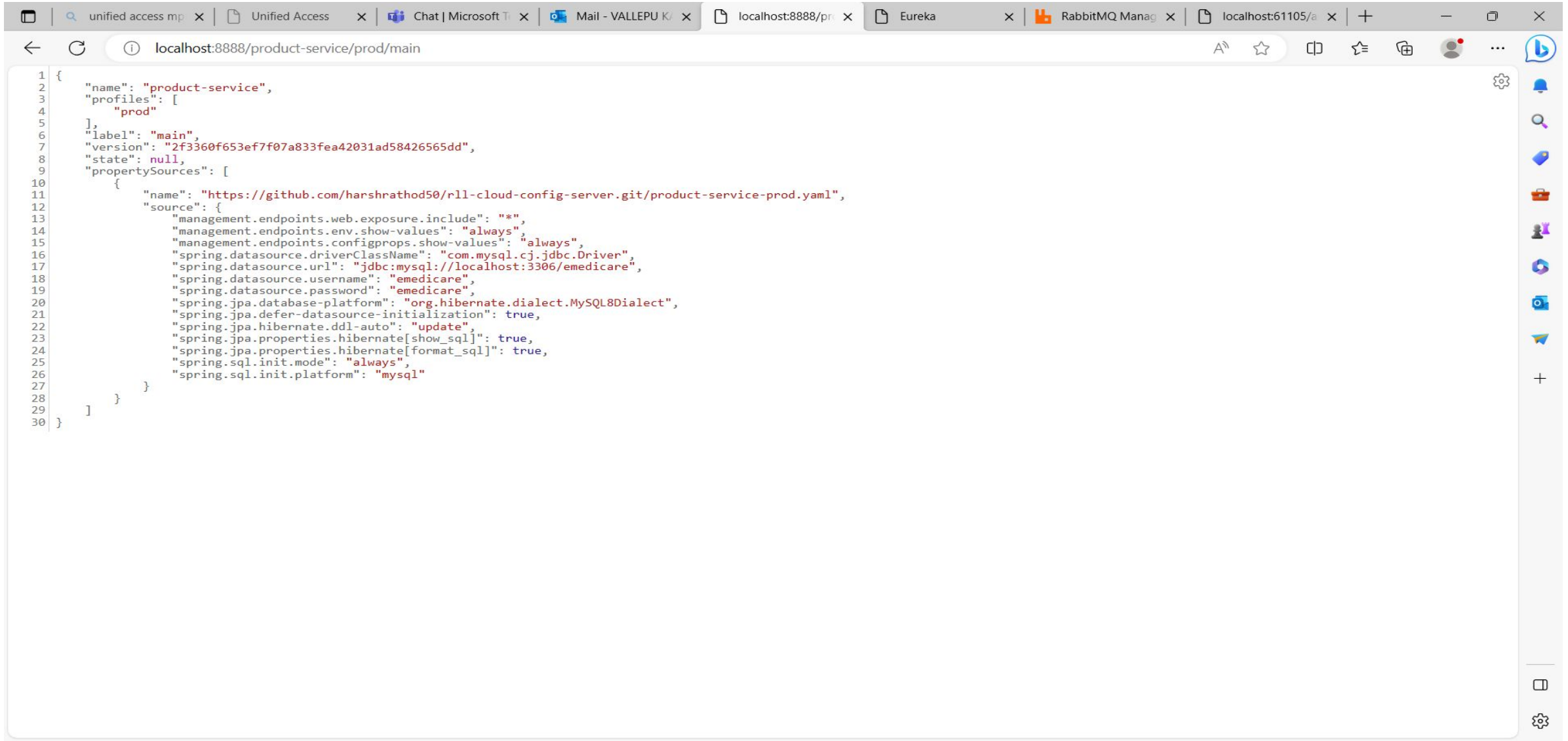
ER Diagram



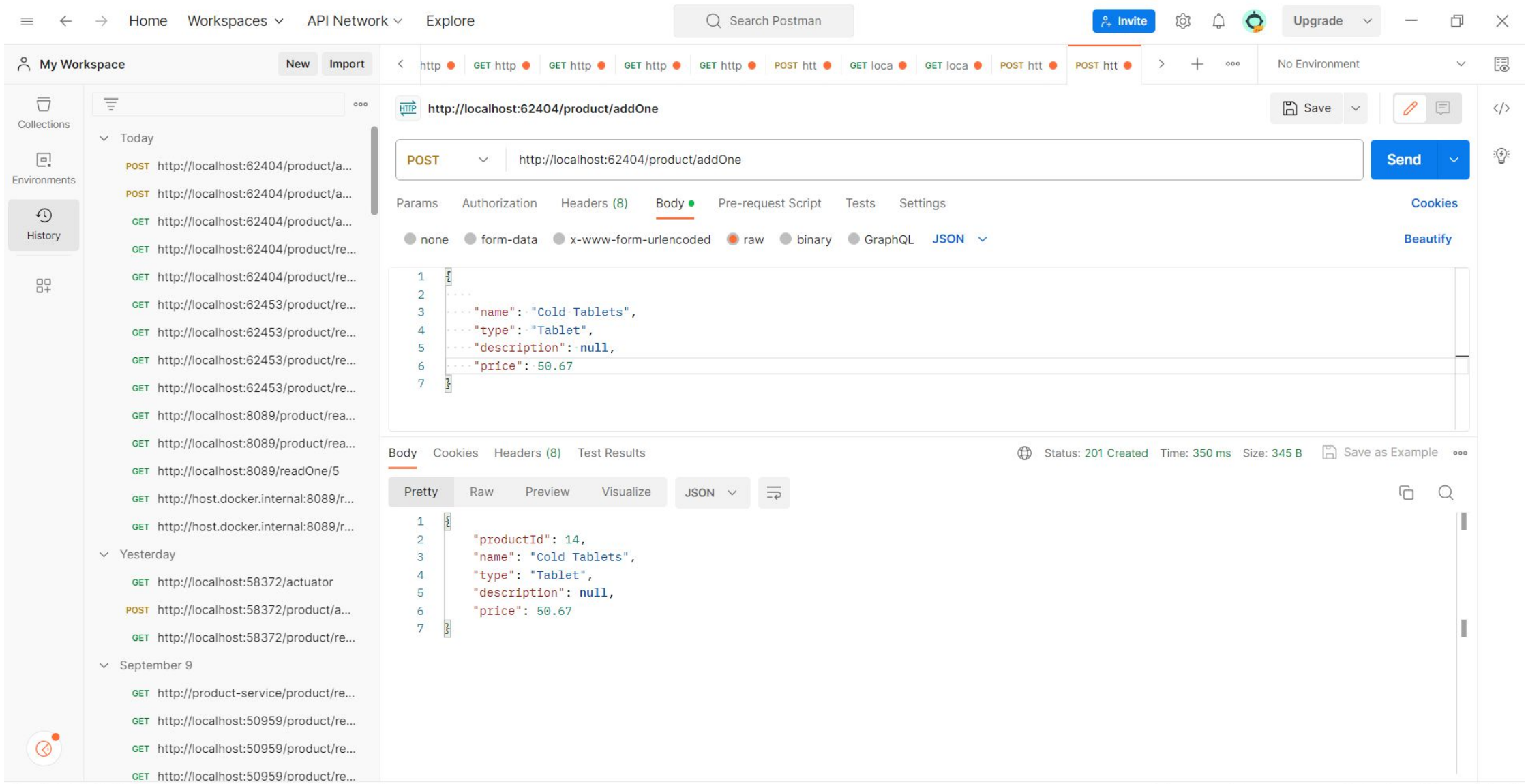
Logic

-CRUD Implementation
-Cloud Config Server
-Eureka Server

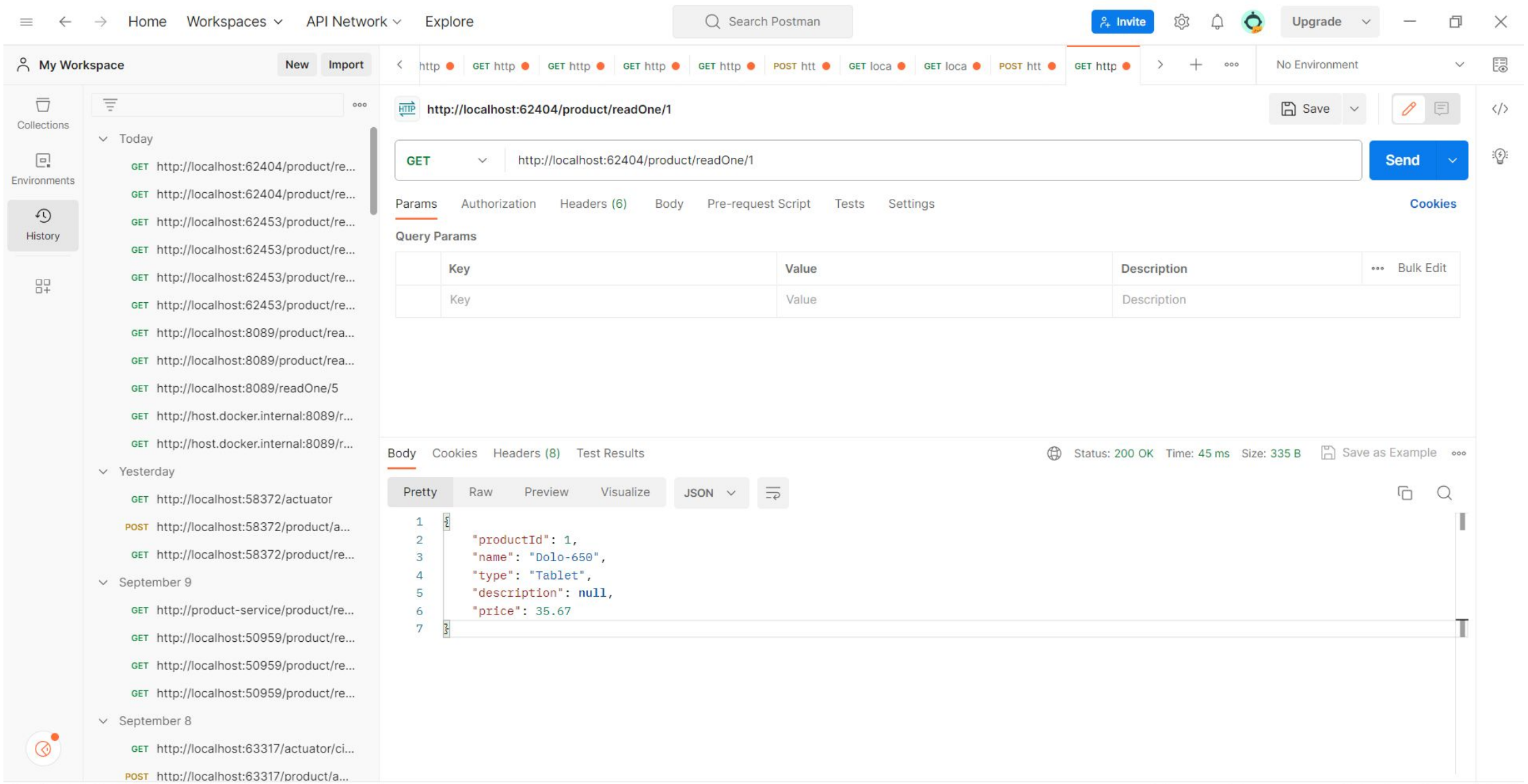
Cloud Config Server



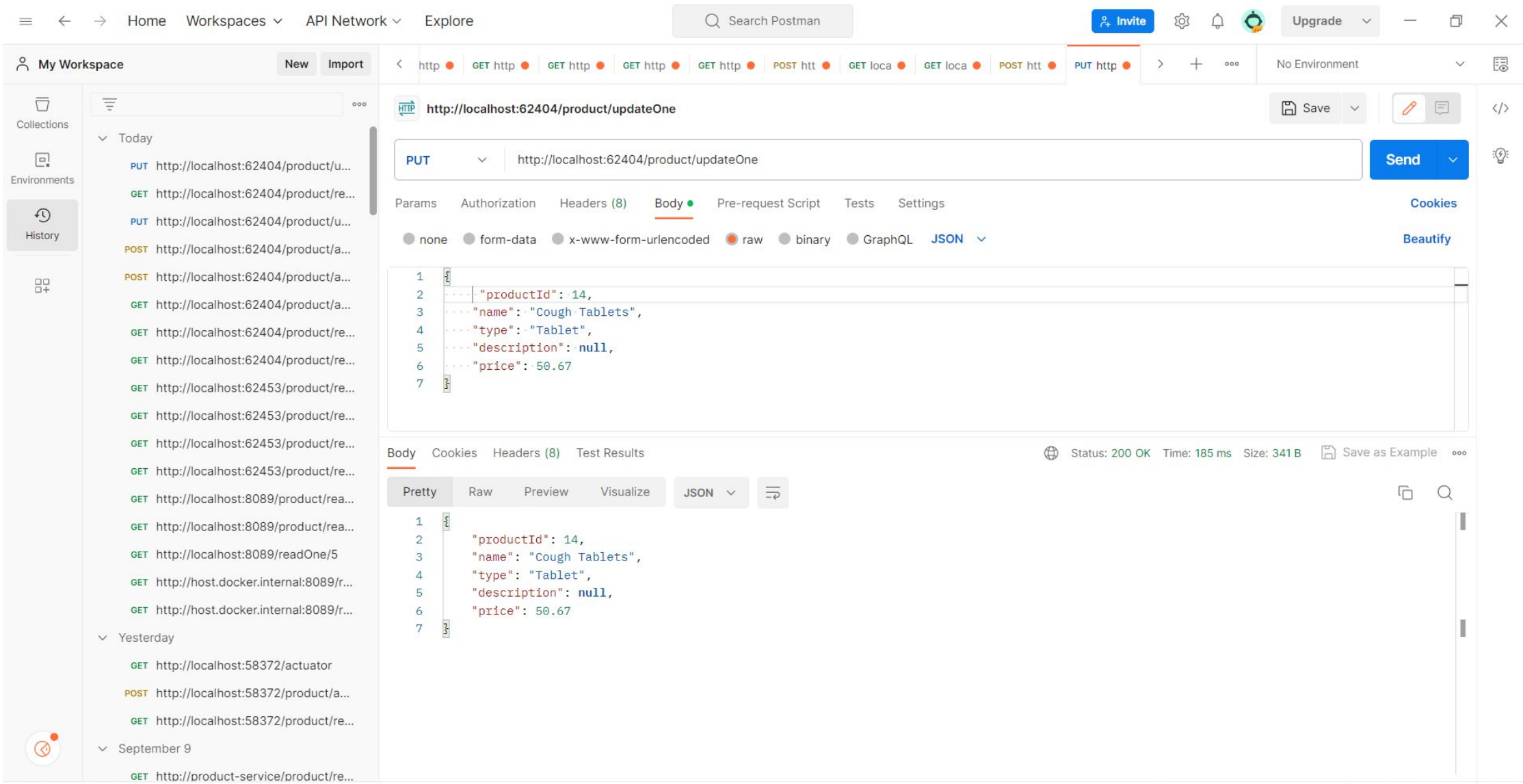
Add Product


Home
Workspaces
API Network
Explore
Search Postman
Invite
Upgrade
My Workspace
New
Import
No Environment
Collections
Environments
History
Today
Yesterday
September 9
http://localhost:62404/product/addOne
Save
POST
http://localhost:62404/product/addOne
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"name": "Cold Tablets",
"type": "Tablet",
"description": null,
"price": 50.67
}
Body
Cookies
Headers (8)
Test Results
Status: 201 Created
Time: 350 ms
Size: 345 B
Save as Example
Pretty
Raw
Preview
Visualize
JSON
{
"productId": 14,
"name": "Cold Tablets",
"type": "Tablet",
"description": null,
"price": 50.67
}
Online
Find and replace
Console
Runner
Capture requests
Cookies
Trash

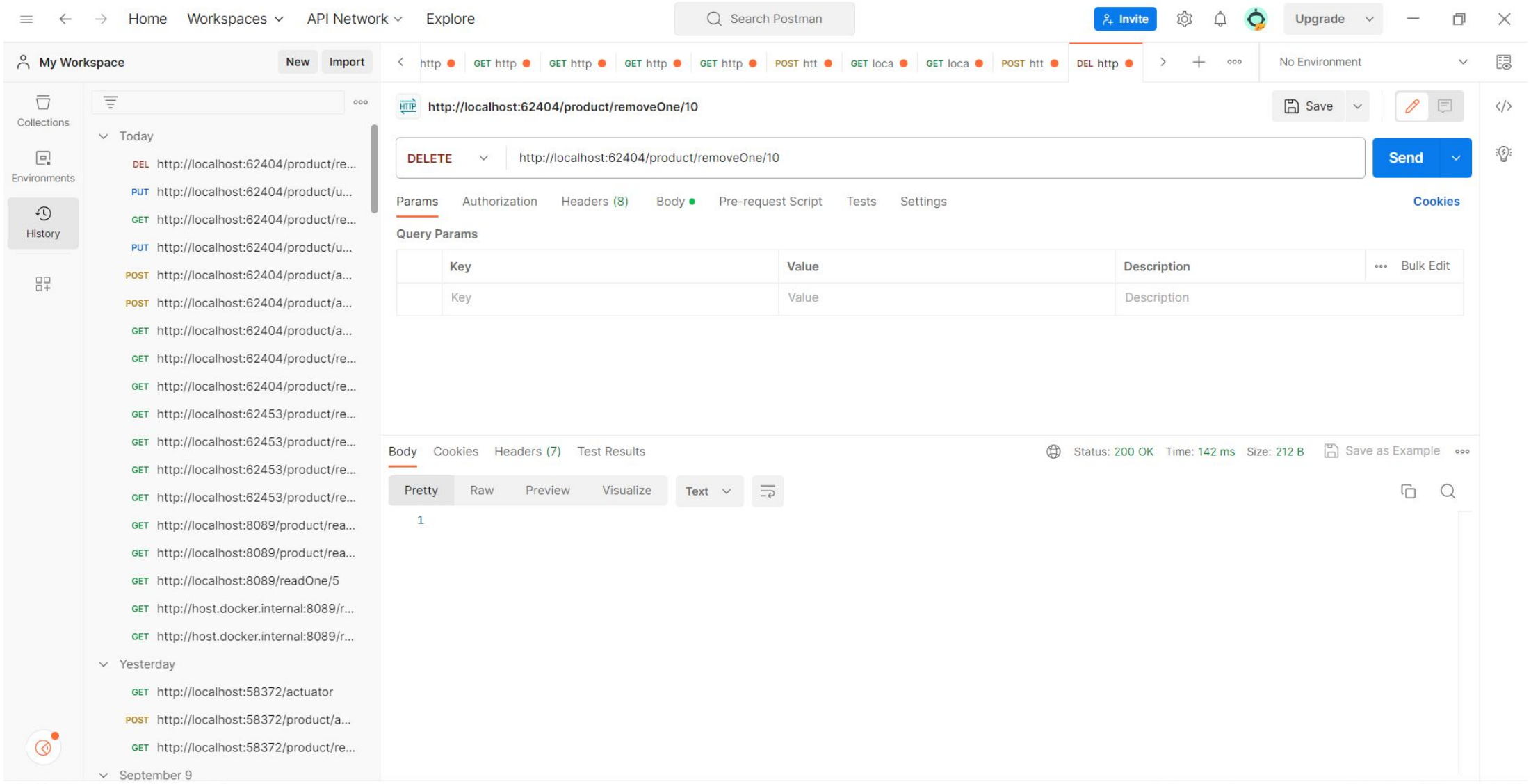
Read Product



Update Product



Remove Product



Users Management

Pavan Kumar

- User management module handles user list related operations.
- User management services relies on user register services for user data.

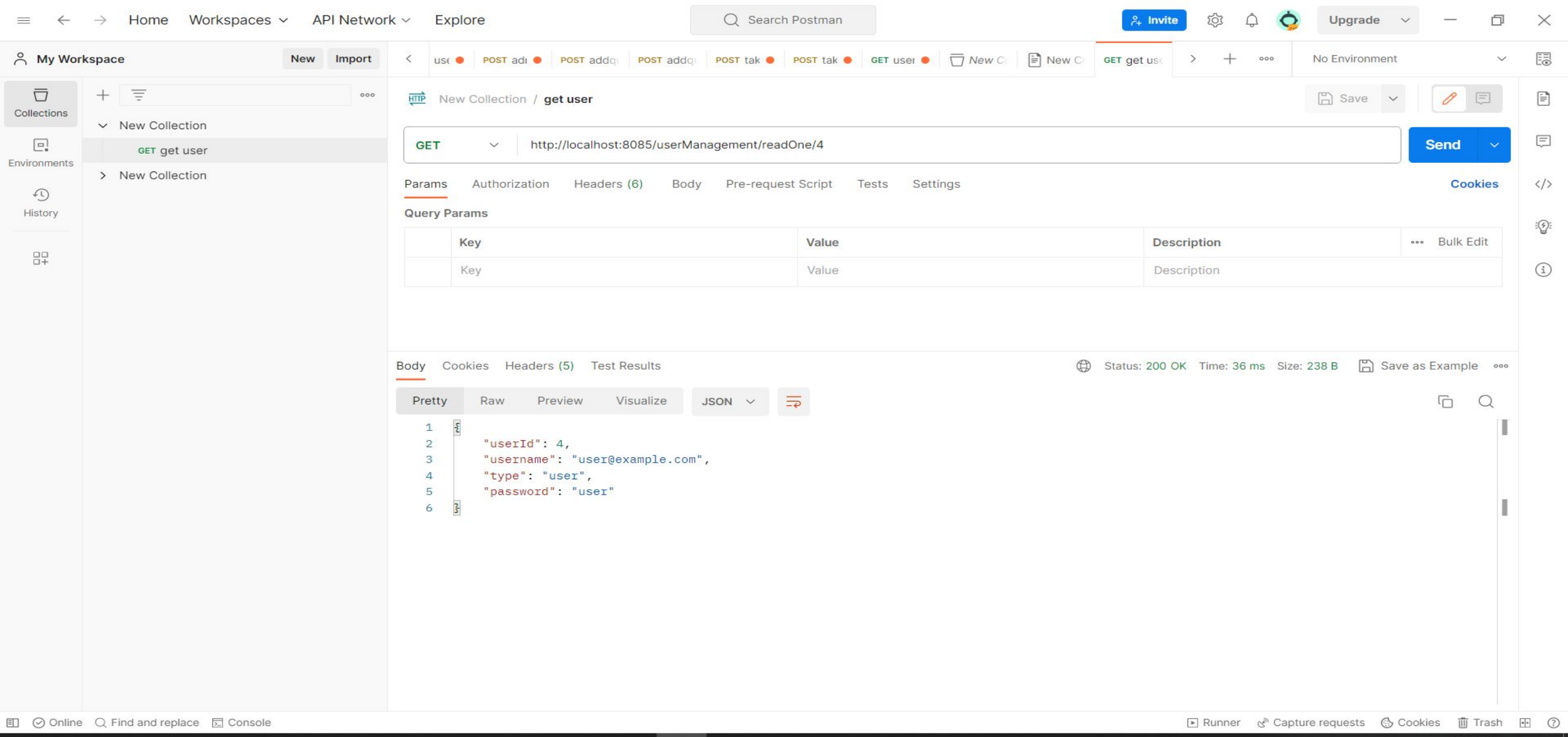
Cloud Services

-config server

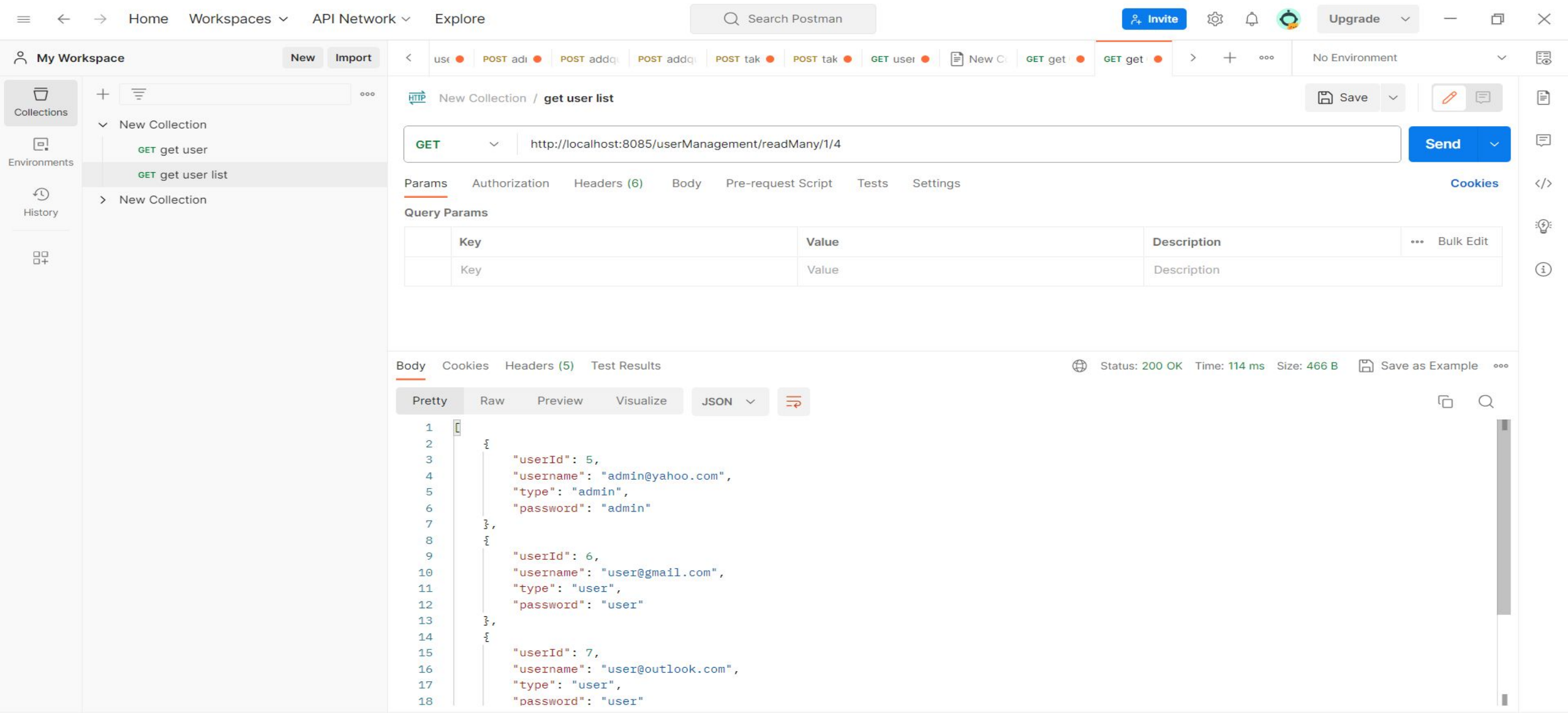
-eureka server

- OpenFeign
- Reselience4j
- Retry

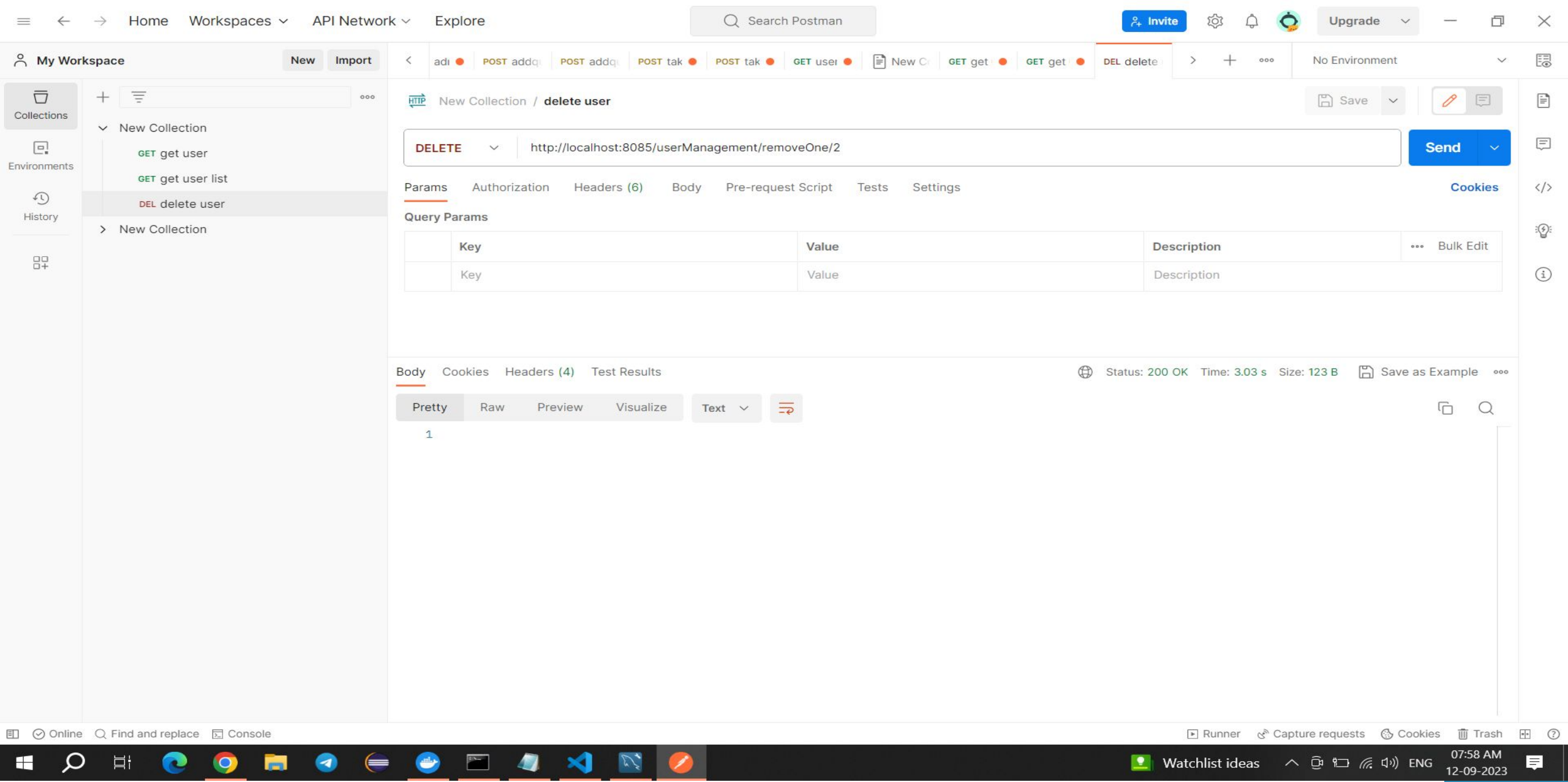
View User


Home
Workspaces
API Network
Explore
Search Postman
Invite
Upgrade
My Workspace
New
Import
No Environment
Collections
Environments
History
New Collection
GET get user
New Collection
New Collection / get user
Save
GET
http://localhost:8085/userManagement/readOne/4
Send
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
Body
Cookies
Headers (5)
Test Results
Status: 200 OK
Time: 36 ms
Size: 238 B
Save as Example
Pretty
Raw
Preview
Visualize
JSON
{
"userId": 4,
"username": "user@example.com",
"type": "user",
"password": "user"
}
Online
Find and replace
Console
Runner
Capture requests
Cookies
Trash
28°C Mostly cloudy
ENG
07:54 AM
12-09-2023

Users list


Home
Workspaces
API Network
Explore
Search Postman
Invite
Upgrade
My Workspace
New
Import
No Environment
Collections
Environments
History
New Collection
GET get user
GET get user list
New Collection
New Collection / get user list
Save
GET
http://localhost:8085/userManagement/readMany/1/4
Send
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
Body
Cookies
Headers (5)
Test Results
Status: 200 OK
Time: 114 ms
Size: 466 B
Save as Example
Pretty
Raw
Preview
Visualize
JSON
[
{
"userId": 5,
"username": "admin@yahoo.com",
"type": "admin",
"password": "admin"
},
{
"userId": 6,
"username": "user@gmail.com",
"type": "user",
"password": "user"
},
{
"userId": 7,
"username": "user@outlook.com",
"type": "user",
"password": "user"
Online
Find and replace
Console
Runner
Capture requests
Cookies
Trash
Watchlist ideas
ENG
07:56 AM
12-09-2023

Delete user


Home
Workspaces
API Network
Explore
Search Postman
Invite
Upgrade
My Workspace
New
Import
No Environment
Collections
Environments
History
New Collection
GET get user
GET get user list
DEL delete user
New Collection
New Collection / delete user
Save
DELETE
http://localhost:8085/userManagement/removeOne/2
Send
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
Body
Cookies
Headers (4)
Test Results
Status: 200 OK
Time: 3.03 s
Size: 123 B
Save as Example
Pretty
Raw
Preview
Visualize
Text
1
Online
Find and replace
Console
Runner
Capture requests
Cookies
Trash
Watchlist ideas
ENG
07:58 AM
12-09-2023

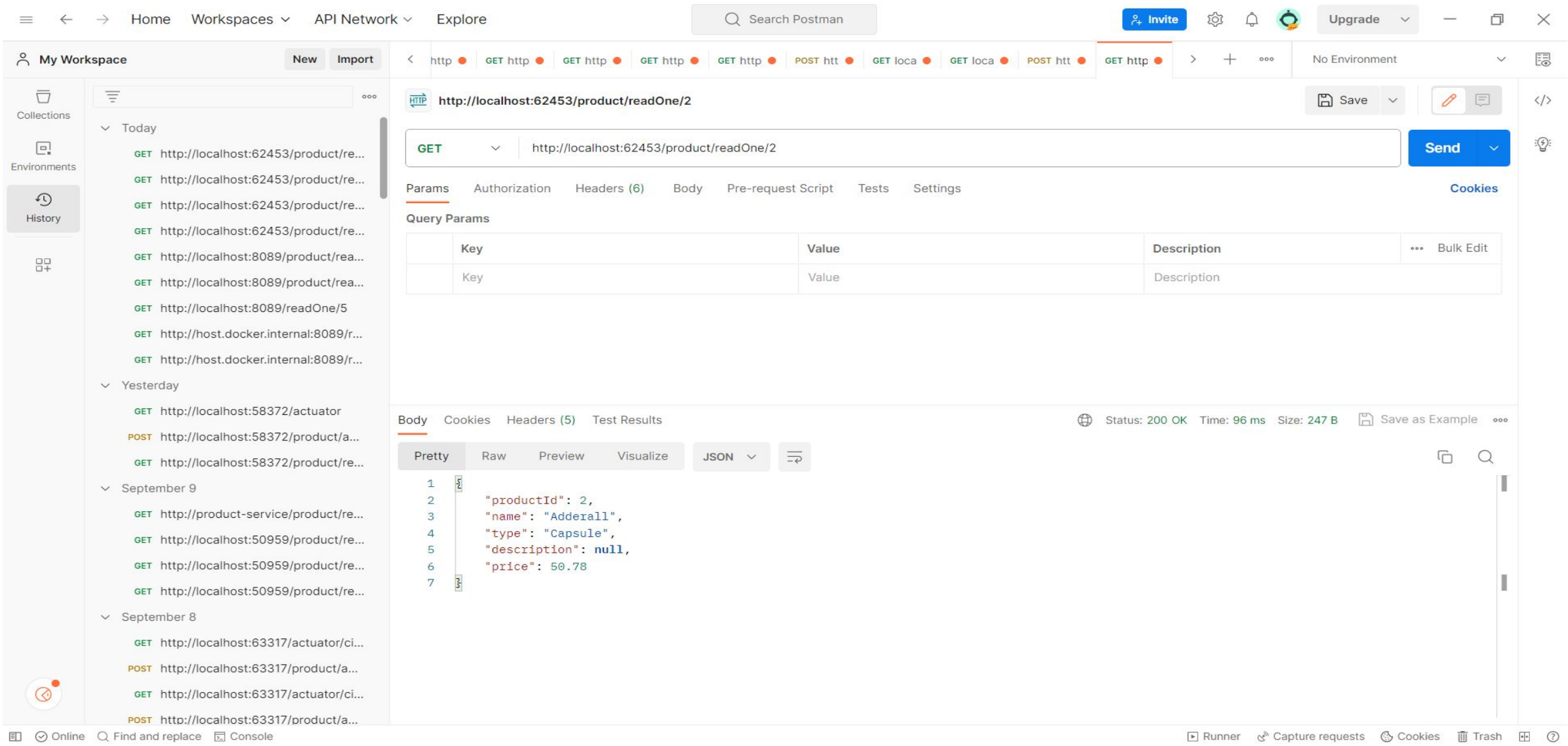
User-Dashboard-Service:

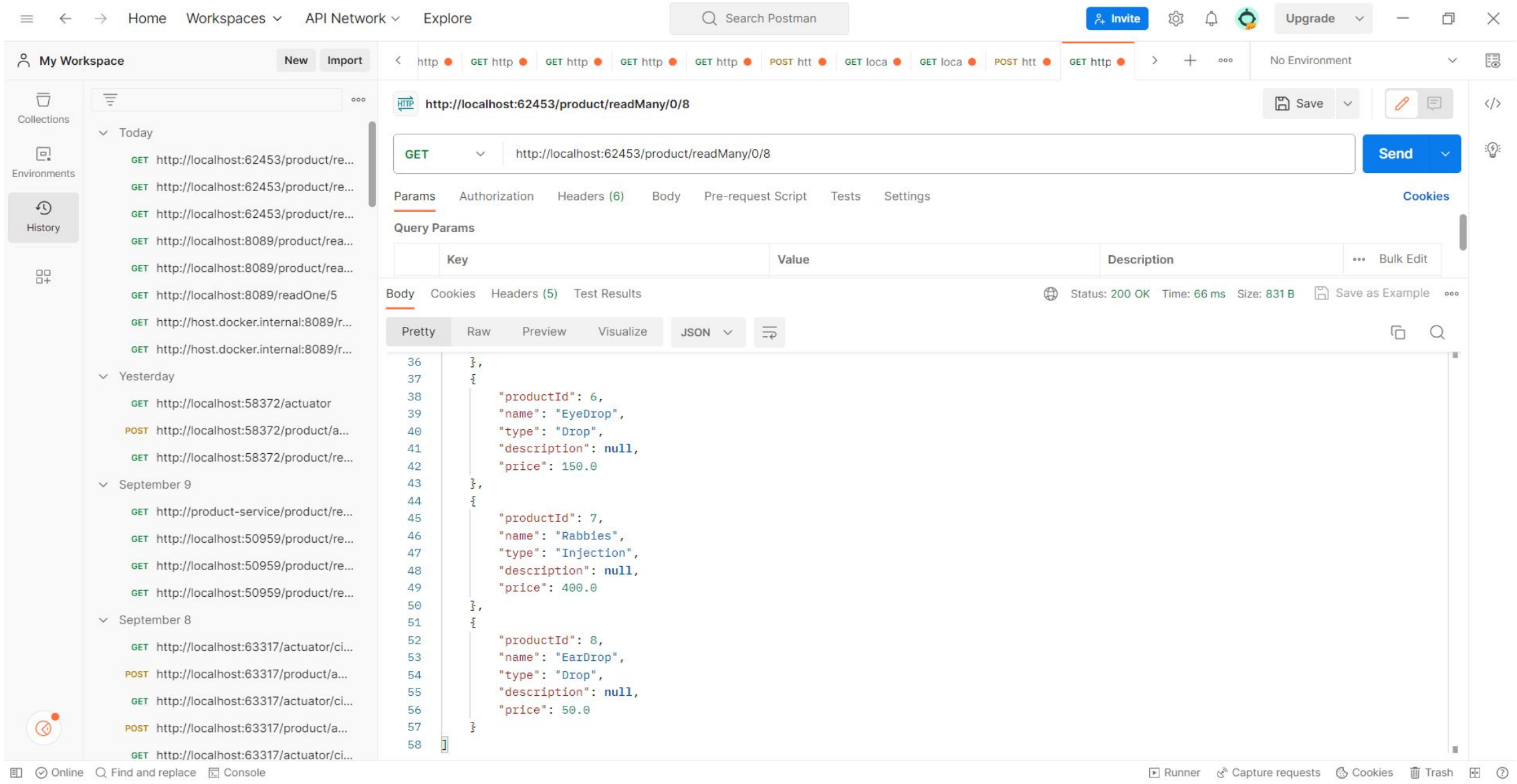
- Logic

  -Feign Client
  -Reselience4j
  -Retry

Search Product

Varshita




Home
Workspaces
API Network
Explore
Search Postman
Invite
Upgrade
My Workspace
New
Import
No Environment
Collections
Environments
History
Today
Yesterday
September 9
September 8
http://localhost:62453/product/readMany/0/8
Save
GET
http://localhost:62453/product/readMany/0/8
Send
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
Body
Cookies
Headers (5)
Test Results
Status: 200 OK
Time: 66 ms
Size: 831 B
Save as Example
Pretty
Raw
Preview
Visualize
JSON
36 },
37 {
38 "productId": 6,
39 "name": "EyeDrop",
40 "type": "Drop",
41 "description": null,
42 "price": 150.0
43 },
44 {
45 "productId": 7,
46 "name": "Rabbies",
47 "type": "Injection",
48 "description": null,
49 "price": 400.0
50 },
51 {
52 "productId": 8,
53 "name": "EarDrop",
54 "type": "Drop",
55 "description": null,
56 "price": 50.0
57 }
58]
Online
Find and replace
Console
Runner
Capture requests
Cookies
Trash

Change password

Rajalaxmi

- Change password services uses Feign for making HTTP requests to “user-register-service” to change a password

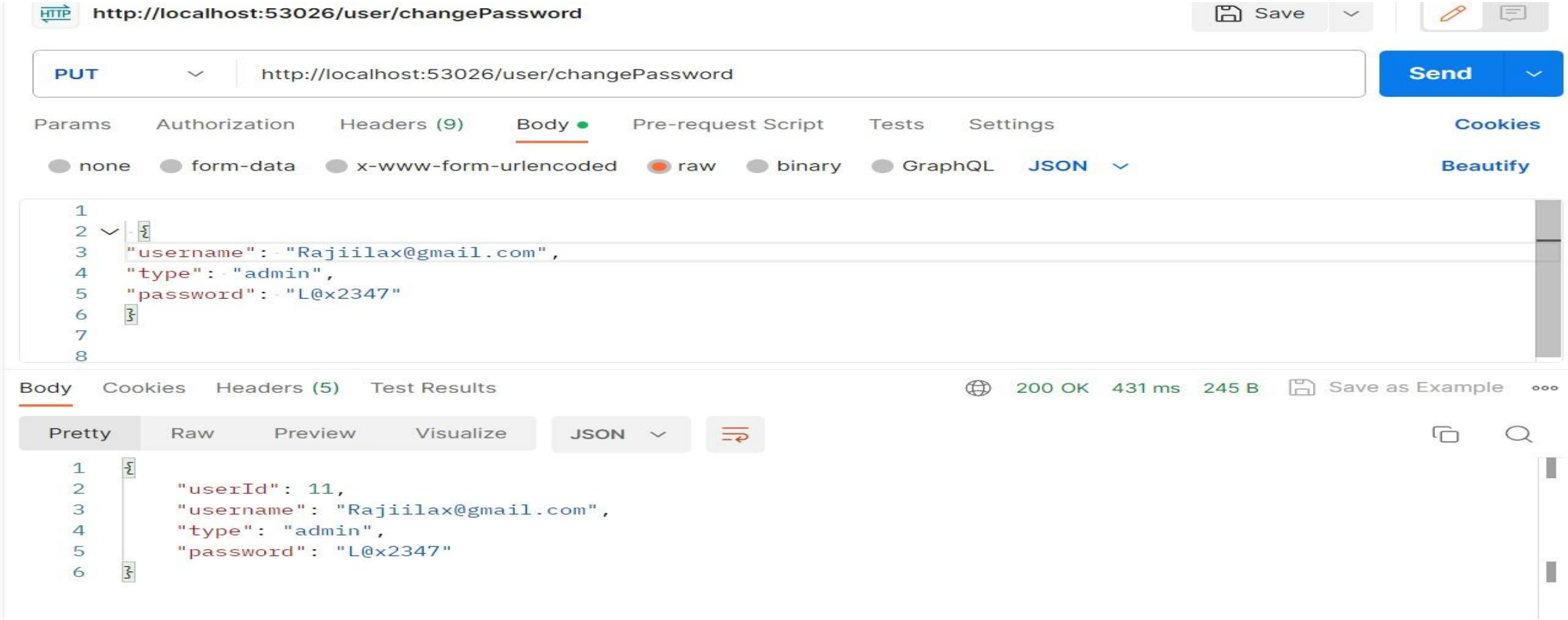
- It also uses the @CircuitBreaker and @Retry annotations for fault tolerance, specifying a fallback method in case of failures.

Cloud Services

- Config
- Eureka
- OpenFeign
- Reselience4j
- Retry

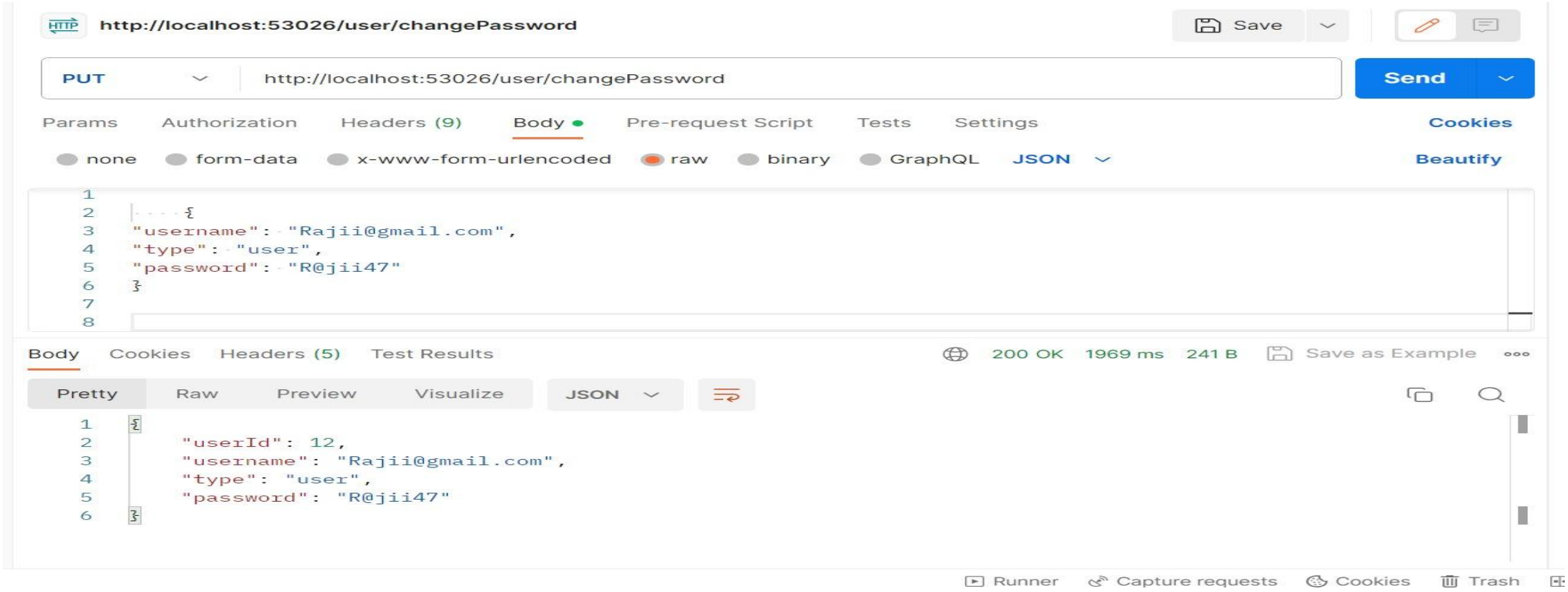
Change password

Admin:



Change password

User:

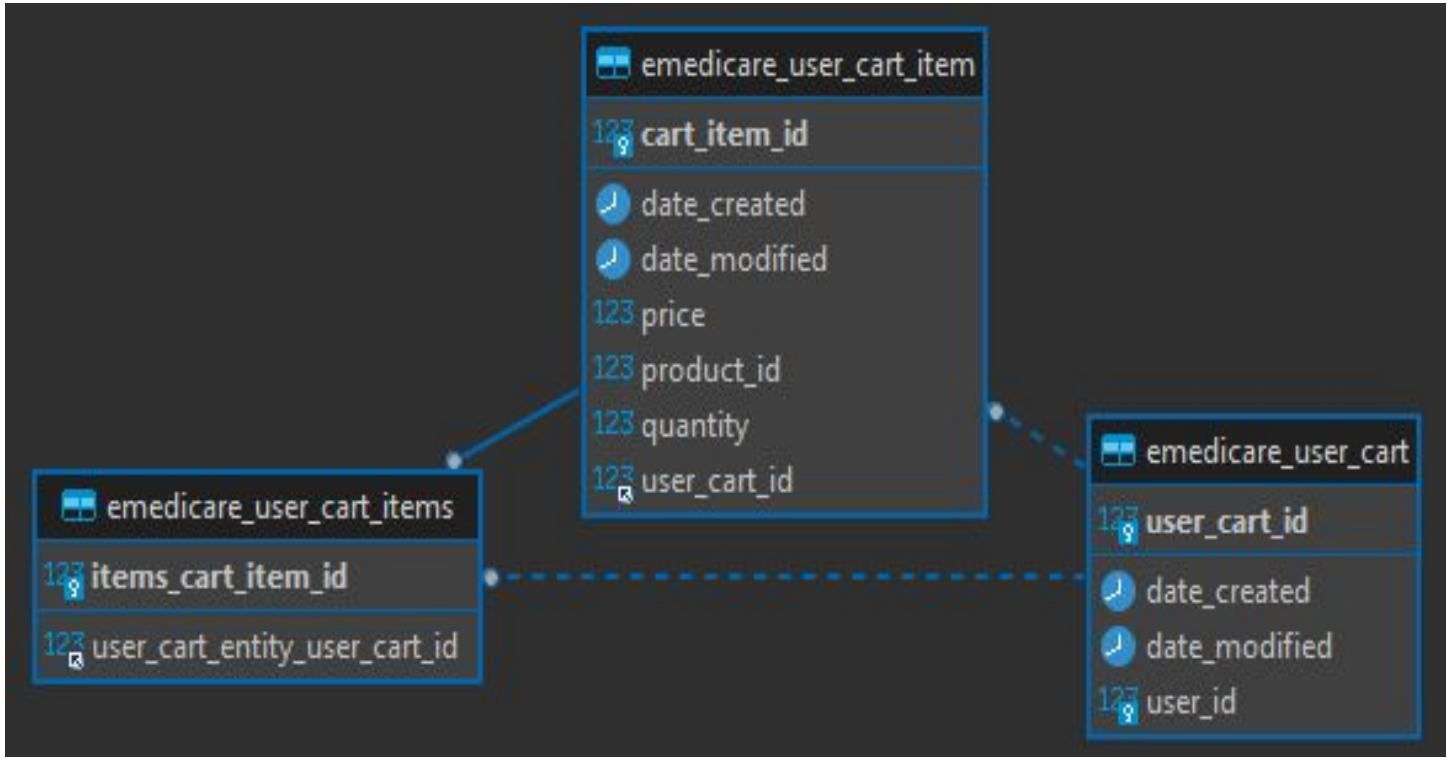


Cart

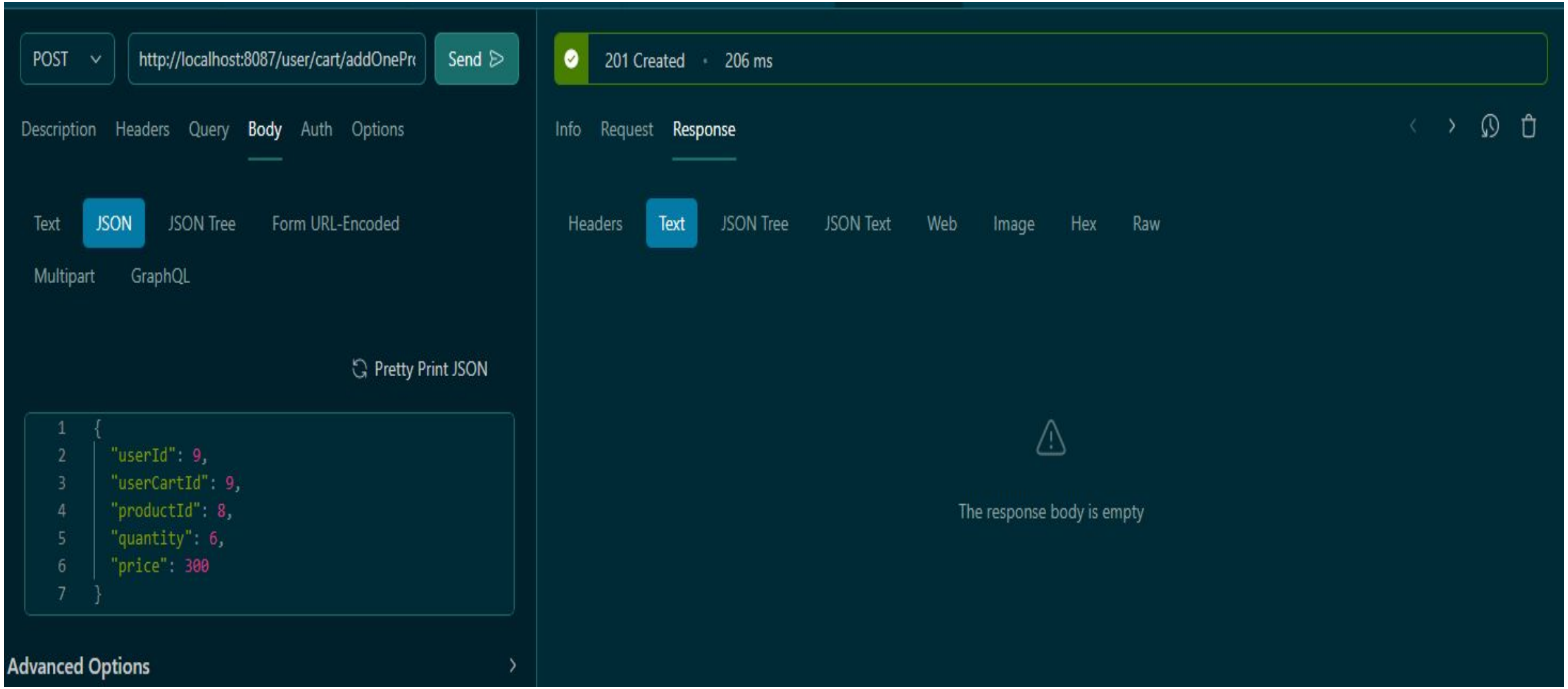
Harsh

Logic:

- Config Server
- Eureka Server
-



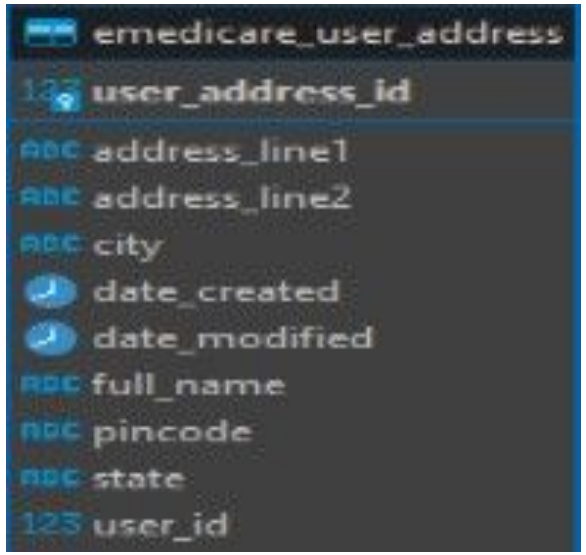
Add product to cart api



USER ADDRESS SERVICE

Logic:

- Config Server
- Eureka Server



Microservice

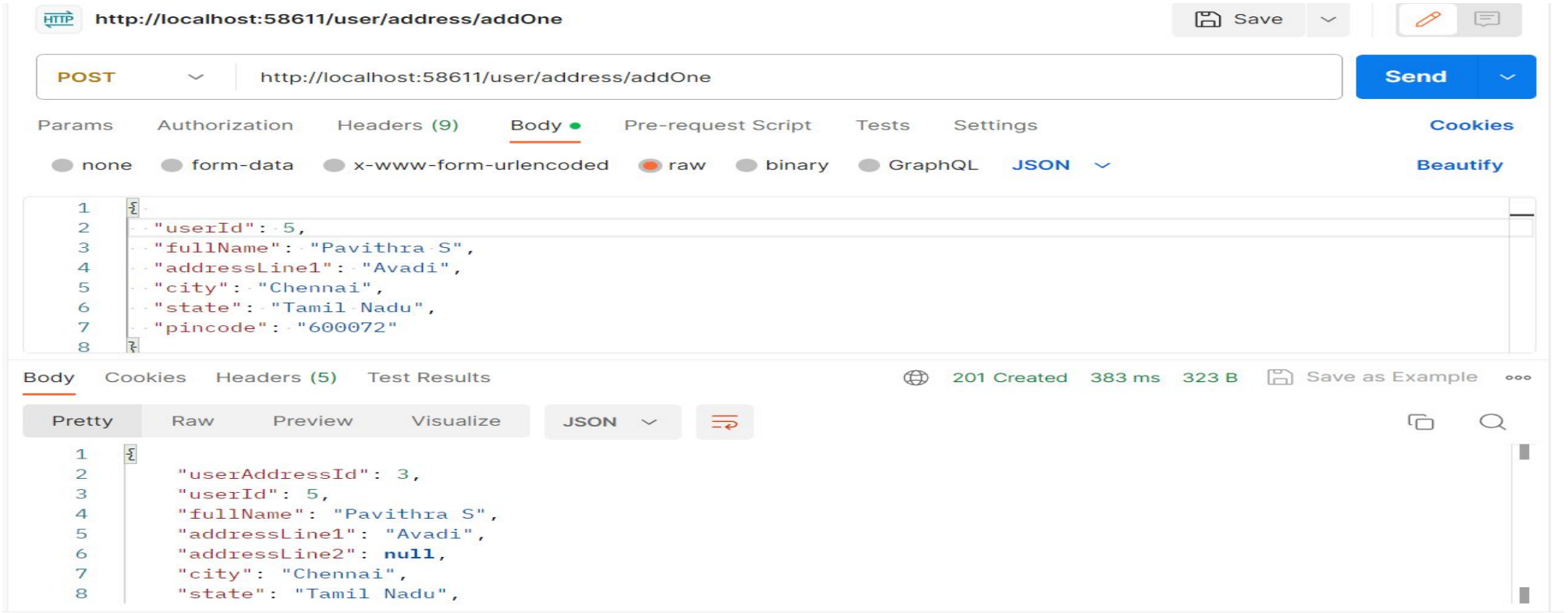
UserAddressController.java

emedicare-user-address-service > src > main > java > com > emedicare > userAddressService > controller > UserAddressCo

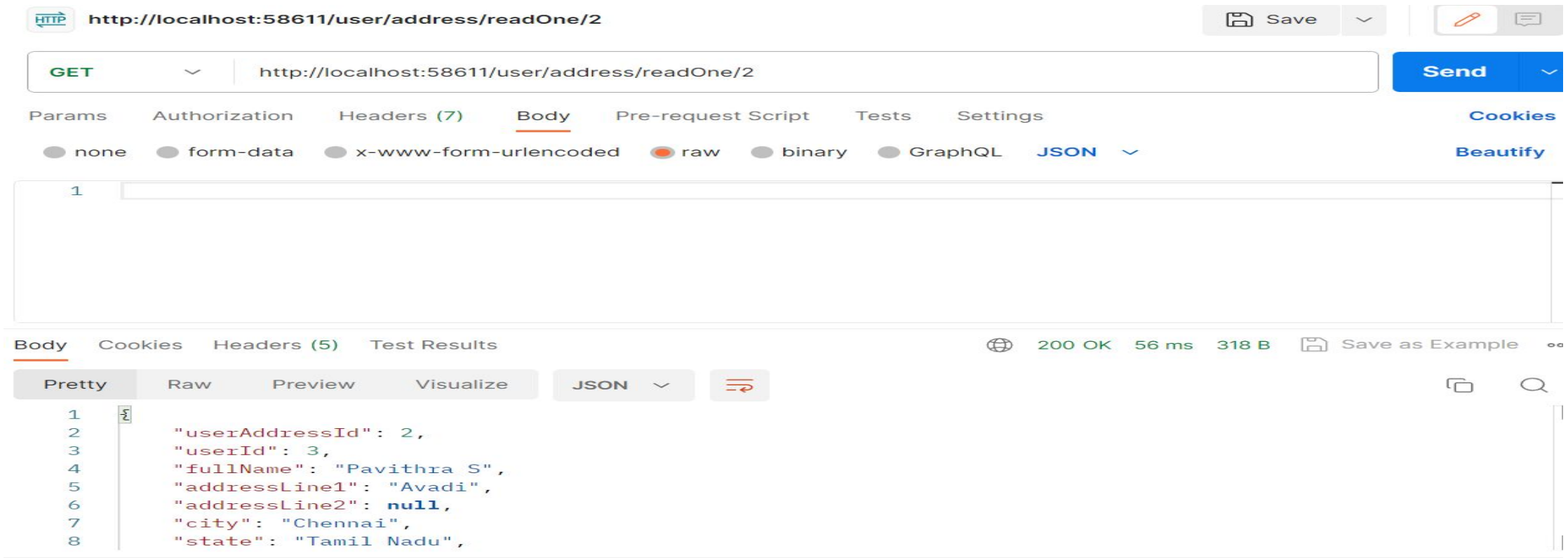
```java
@RestController(value = "UserAddressController")
@Scope(value = "request")
@RequestMapping(value = "/user/address")
public class UserAddressController {
    @Autowired
    @Qualifier(value = "UserAddressService")
    UserAddressService userAddressService;

    You, 7 days ago | 1 author (You)
    @PostMapping(
        value = "/addOne",
        produces = MediaType.APPLICATION_JSON_VALUE,
        consumes = MediaType.APPLICATION_JSON_VALUE
    )
    public ResponseEntity<UserAddressDTO> addOne(@RequestBody UserAddressDTO userAddress) {
        return new ResponseEntity<UserAddressDTO>(
            this.userAddressService.addOne(userAddress),
            HttpStatus.CREATED
        );
    }

    You, 7 days ago | 1 author (You)
    @GetMapping(
        value = "/readOne/{userAddressId}",
        produces = MediaType.APPLICATION_JSON_VALUE
    )
    public ResponseEntity<Object> readOne(
        @PathVariable("userAddressId") Long userAddressId
    ) {
        UserAddressDTO userAddressDTO = this.userAddressService.readOne(userAddressId);
        if (userAddressDTO != null) {
            return new ResponseEntity<Object>(userAddressDTO, HttpStatus.OK);
        }
        return new ResponseEntity<Object>(headers:null, HttpStatus.NOT_FOUND);
    }
```

ADD ADDRESS

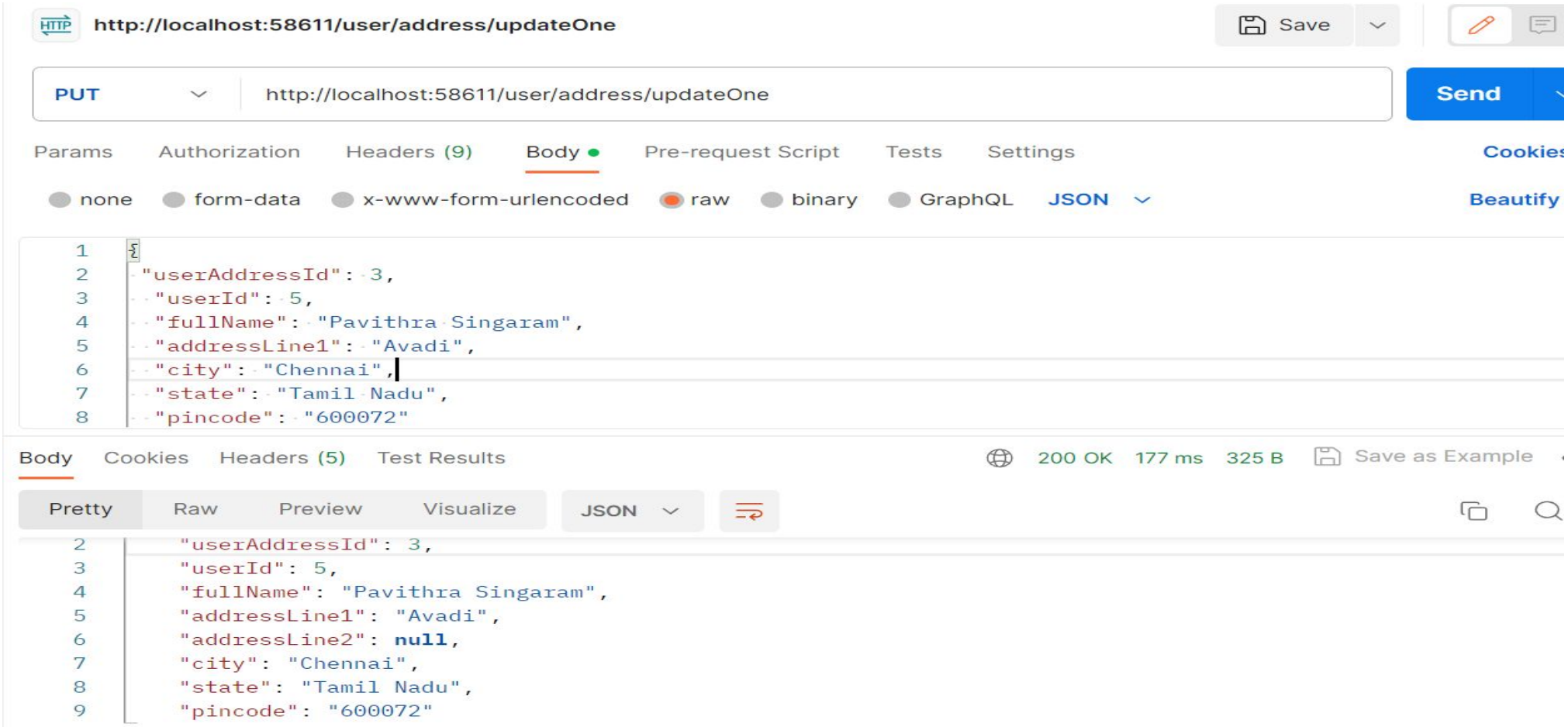
Pavithra



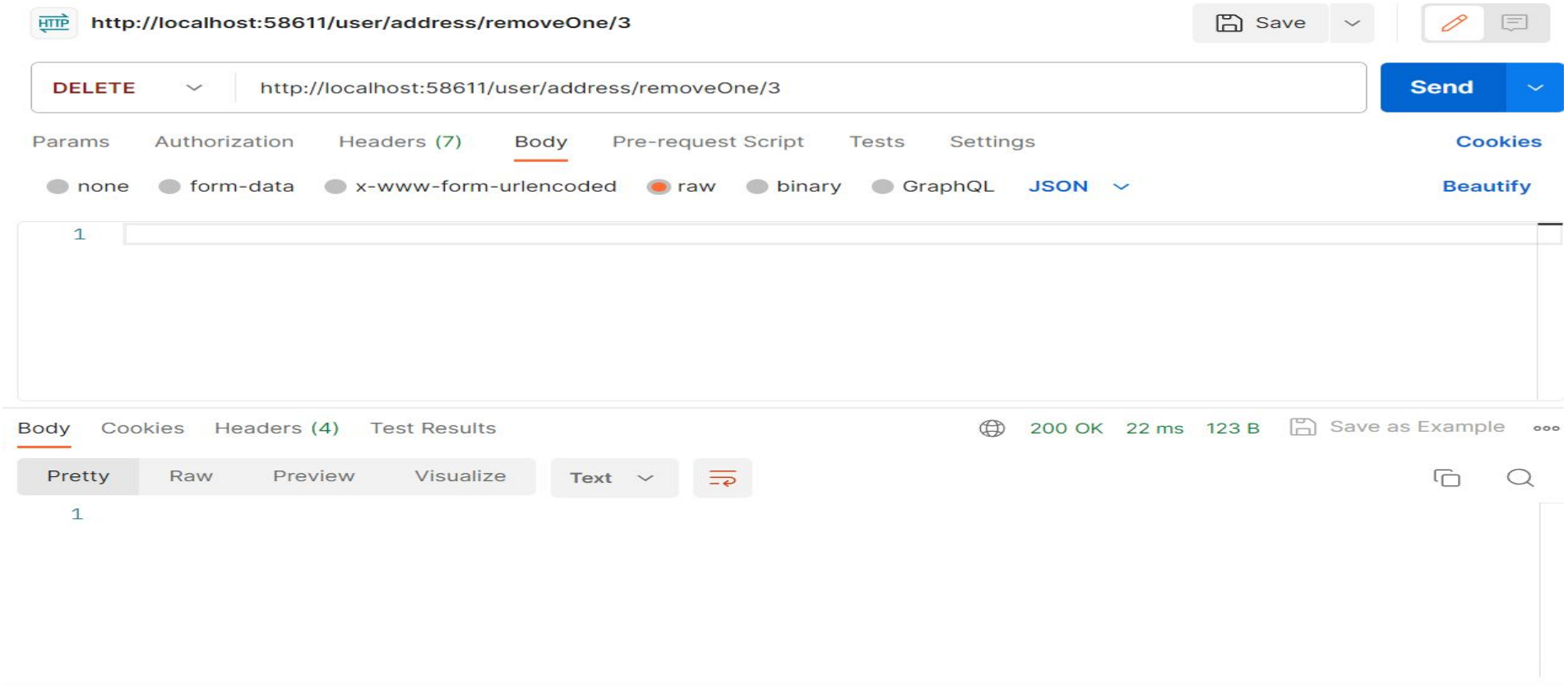
READ ADDRESS



UPDATE ADDRESS



DELETE ADDRESS



Payment

Priya

Logic

Main part:

1.Create microService

2.Connect with Config and Eureka Server

3.Feign Client

4.Reselience4j

5.Retry

5.Zipkin

Table

Table: emedicare_payment

Columns:

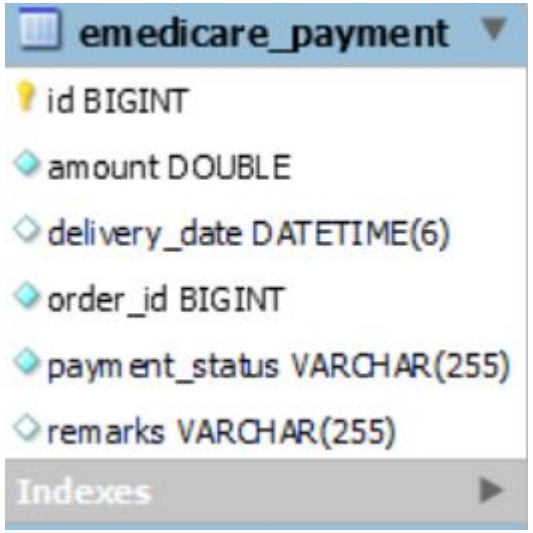
id	bigint AI PK
amount	double
delivery_date	datetime(6)
order_id	bigint
payment_status	varchar(255)
remarks	varchar(255)

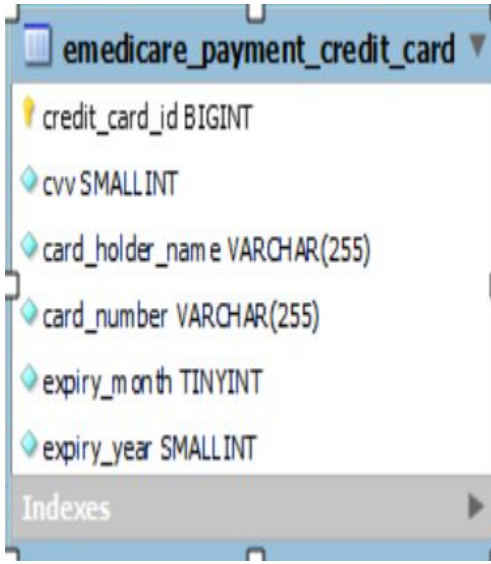
Table:
emedicare_payment_credit_card

Columns:

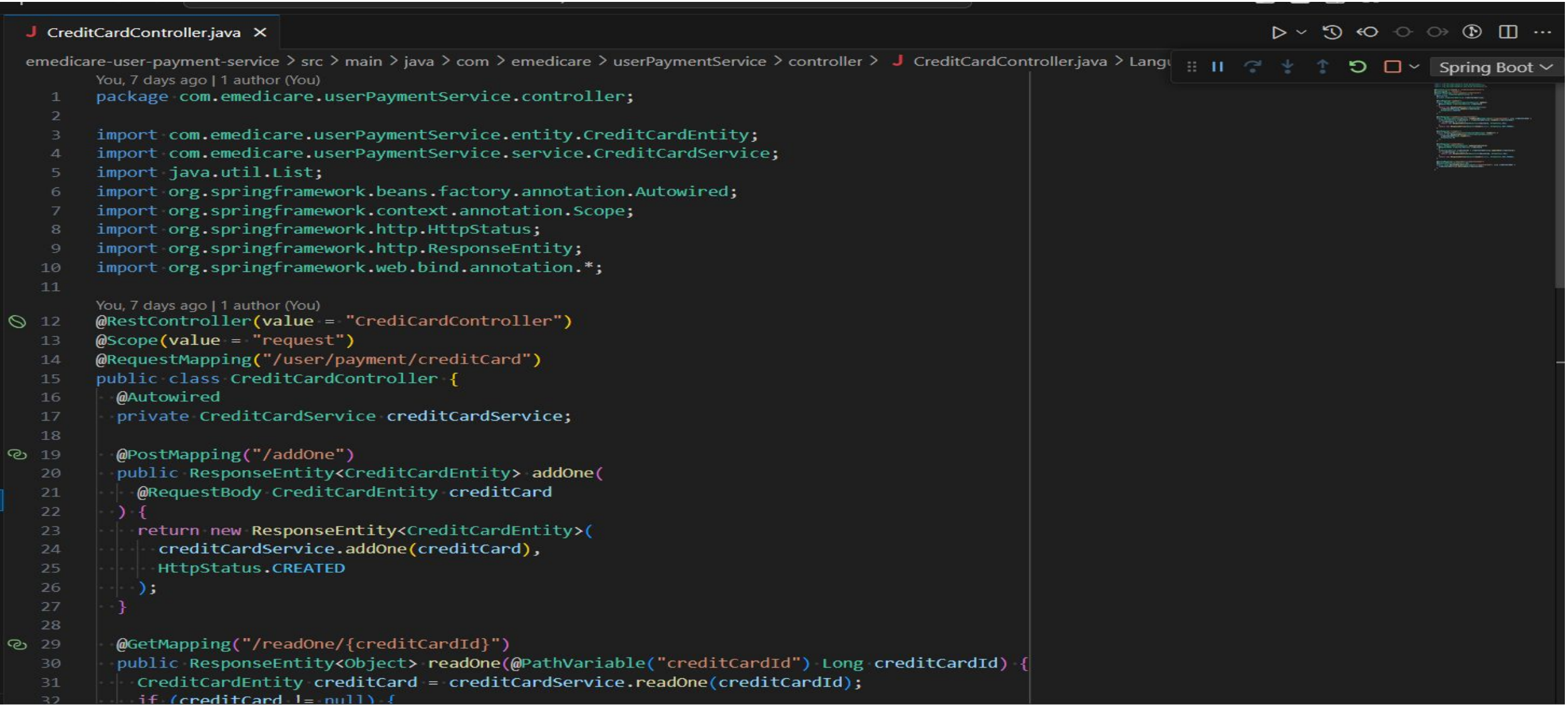
credit_card_id	bigint AI PK
cvv	smallint
card_holder_name	varchar(255)
card_number	varchar(255)
expiry_month	tinyint
expiry_year	smallint

ER Diagram

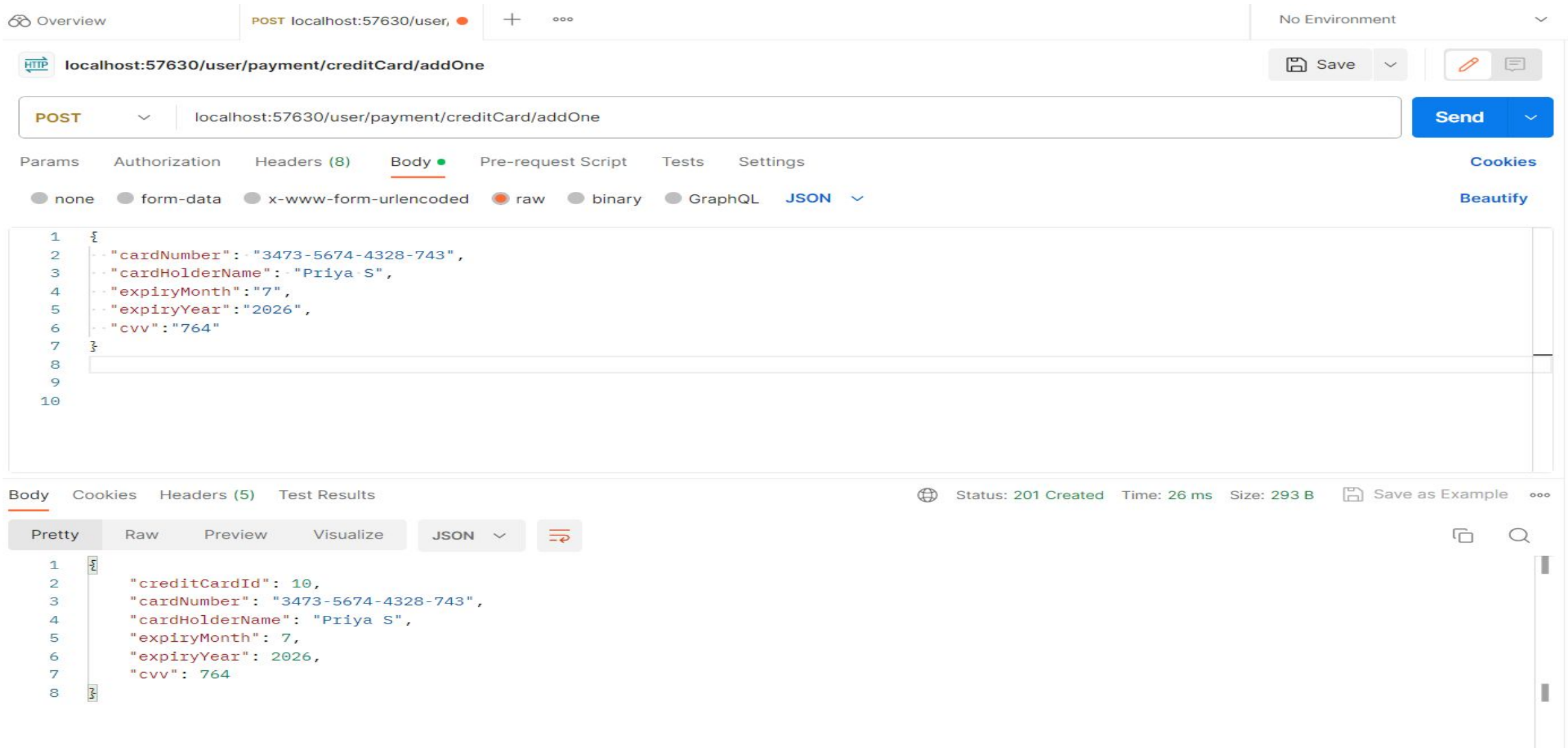




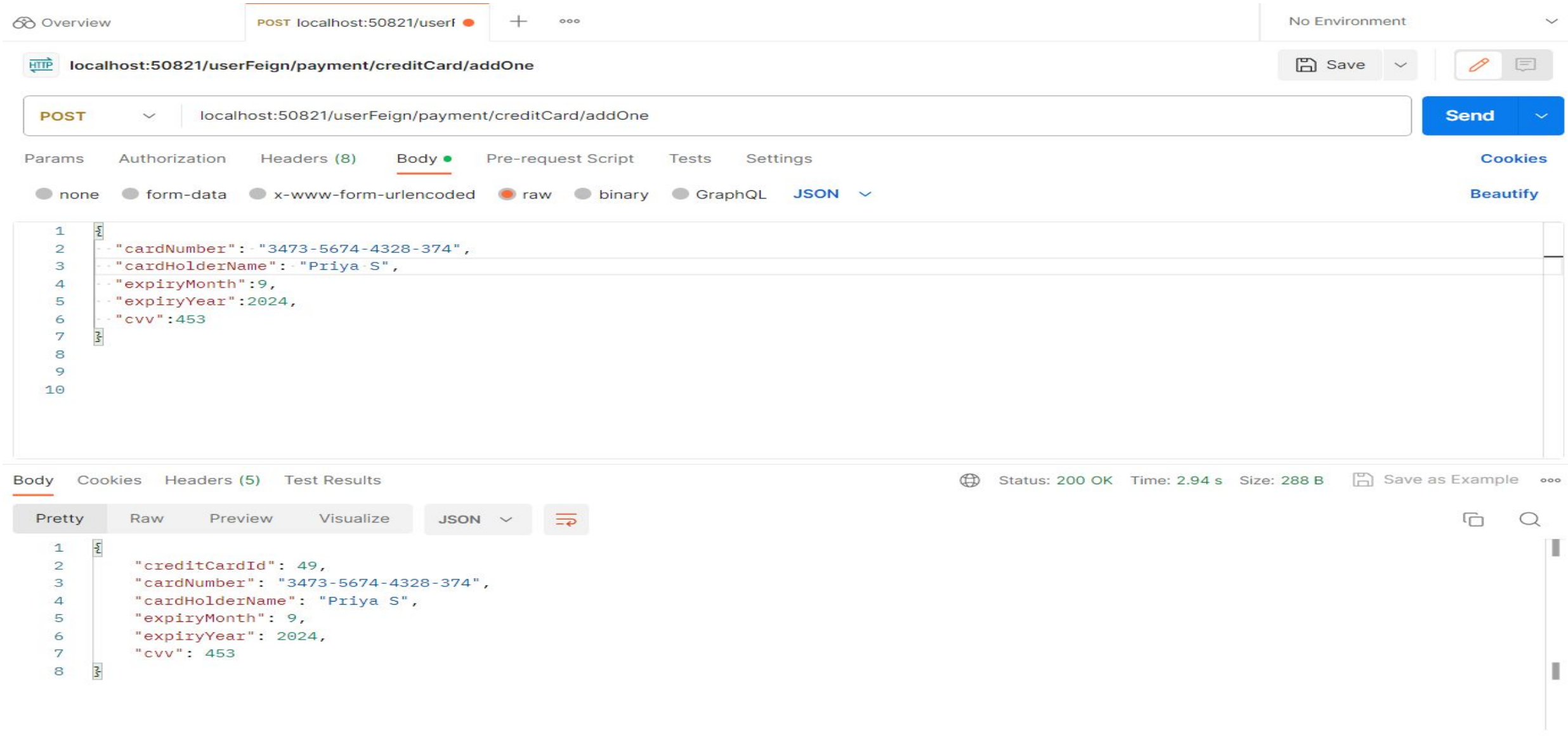
Microservices



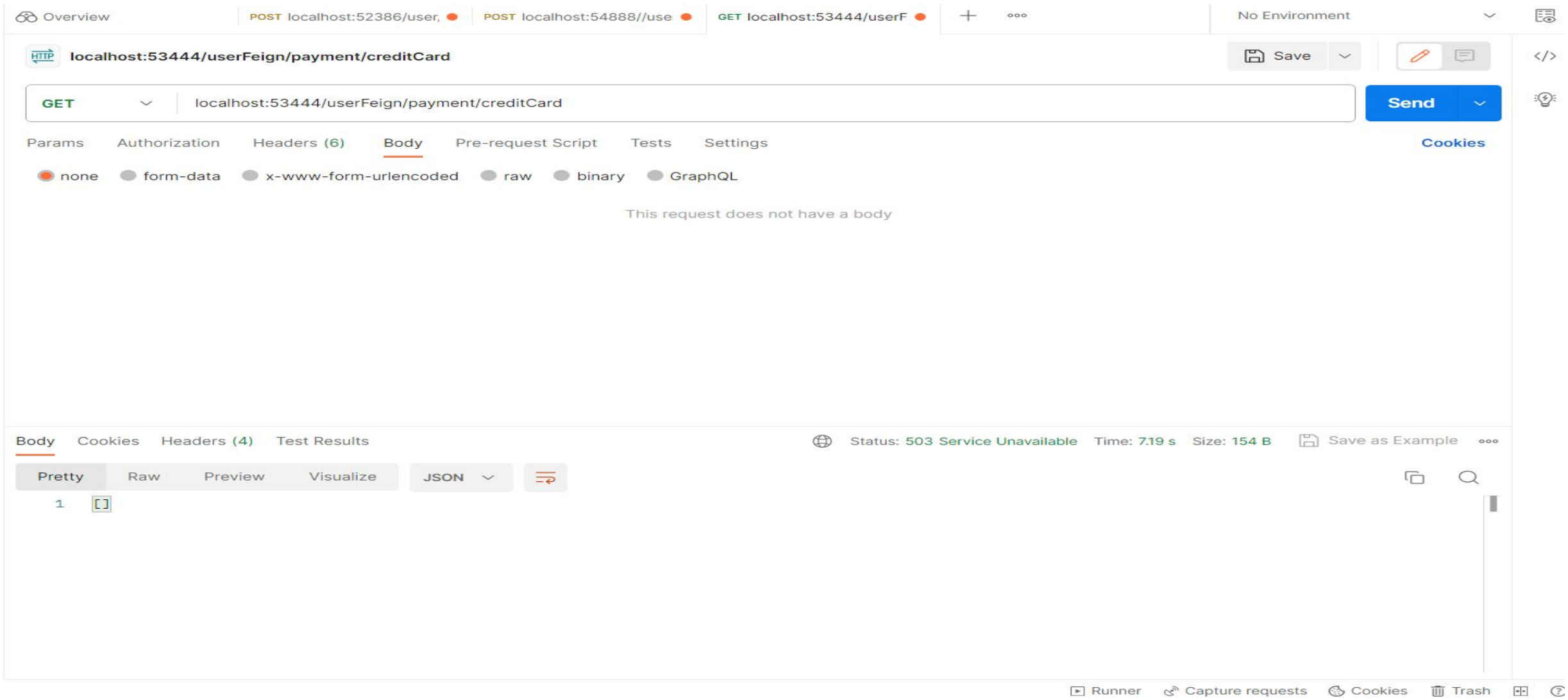
PAYMENT SERVICE



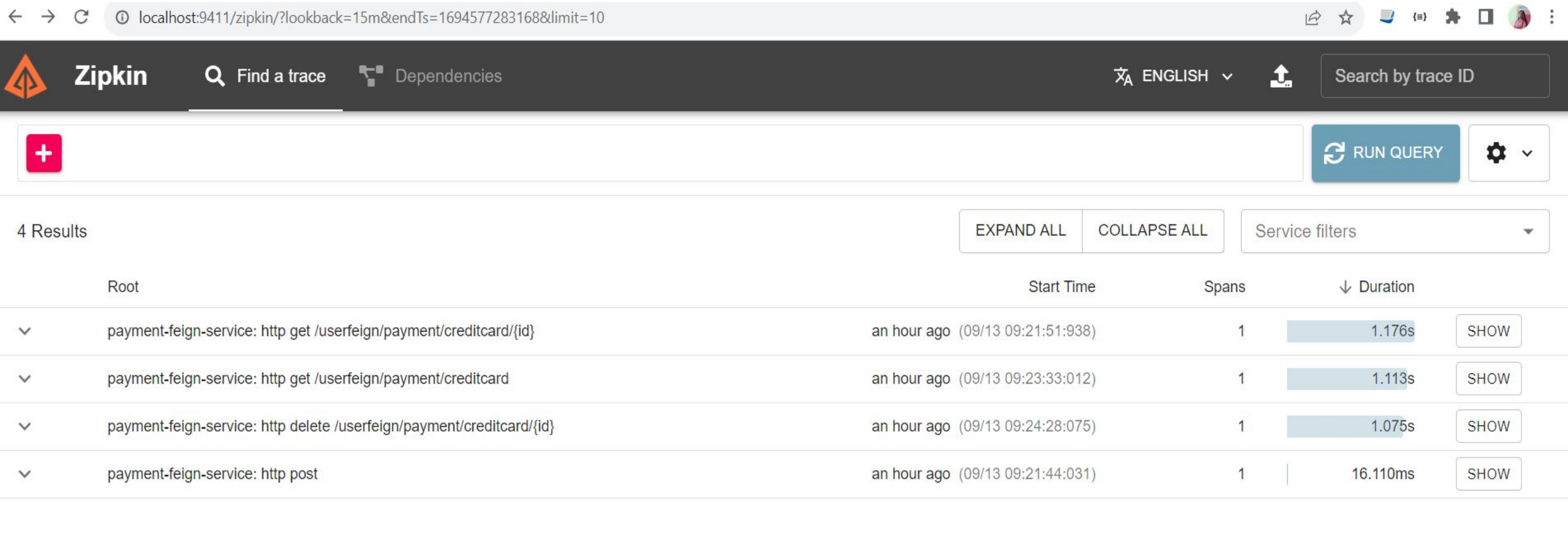
PAYMENT FEIGN SERVICE



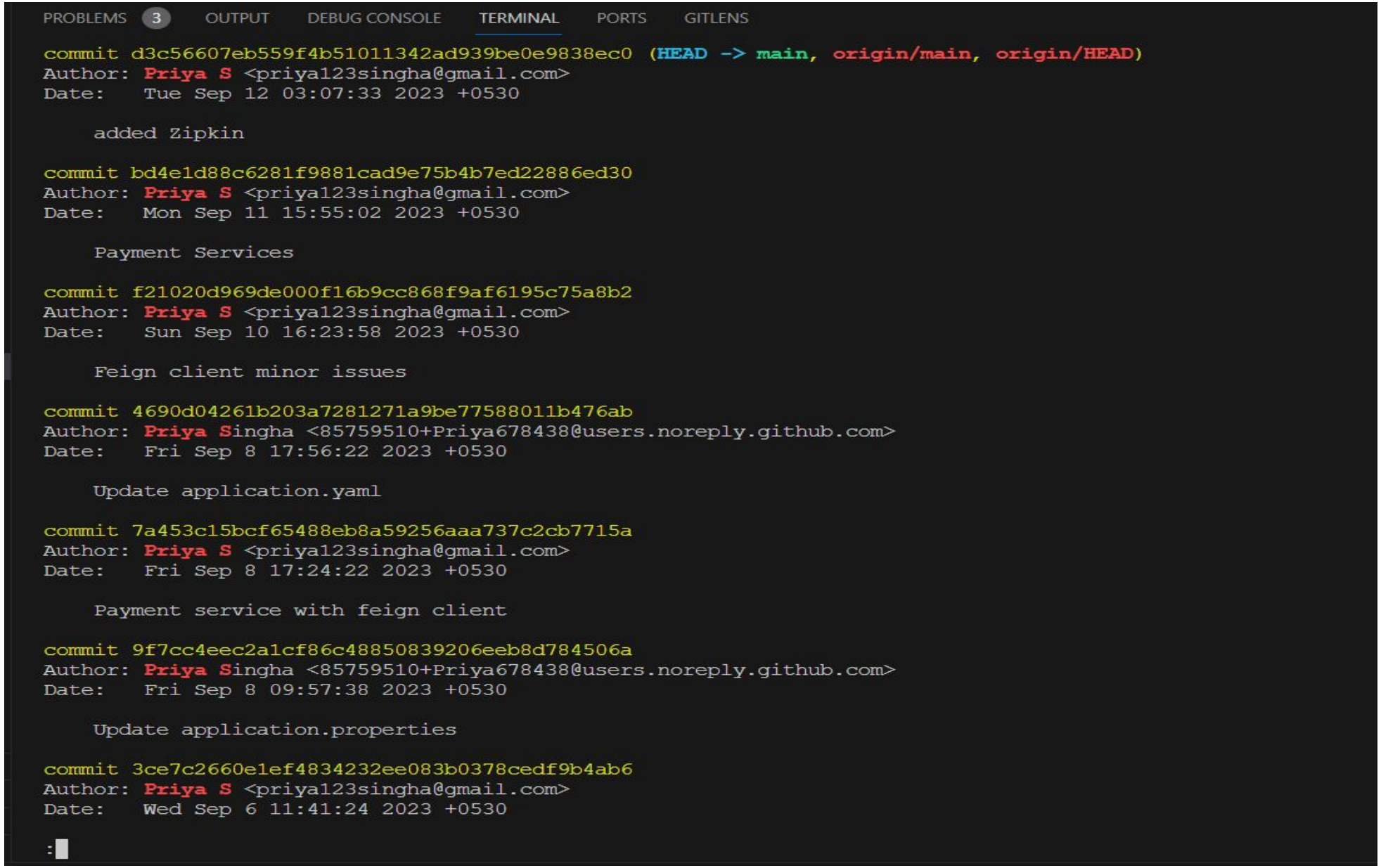
Fallback


Overview
POST localhost:52386/user,
POST localhost:54888//use
GET localhost:53444/userF
No Environment
localhost:53444/userFeign/payment/creditCard
Save
GET
localhost:53444/userFeign/payment/creditCard
Send
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
This request does not have a body
Body
Cookies
Headers (4)
Test Results
Status: 503 Service Unavailable
Time: 7.19 s
Size: 154 B
Save as Example
Pretty
Raw
Preview
Visualize
JSON
1
[]
Runner
Capture requests
Cookies
Trash

Zipkin


localhost:9411/zipkin/?lookback=15m&endTs=1694577283168&limit=10
Zipkin
Find a trace
Dependencies
ENGLISH
Search by trace ID
RUN QUERY
4 Results
EXPAND ALL
COLLAPSE ALL
Service filters
Root
Start Time
Spans
Duration
payment-feign-service: http get /userfeign/payment/creditcard/{id}
an hour ago (09/13 09:21:51:938)
1
1.176s
SHOW
payment-feign-service: http get /userfeign/payment/creditcard
an hour ago (09/13 09:23:33:012)
1
1.113s
SHOW
payment-feign-service: http delete /userfeign/payment/creditcard/{id}
an hour ago (09/13 09:24:28:075)
1
1.075s
SHOW
payment-feign-service: http post
an hour ago (09/13 09:21:44:031)
1
16.110ms
SHOW

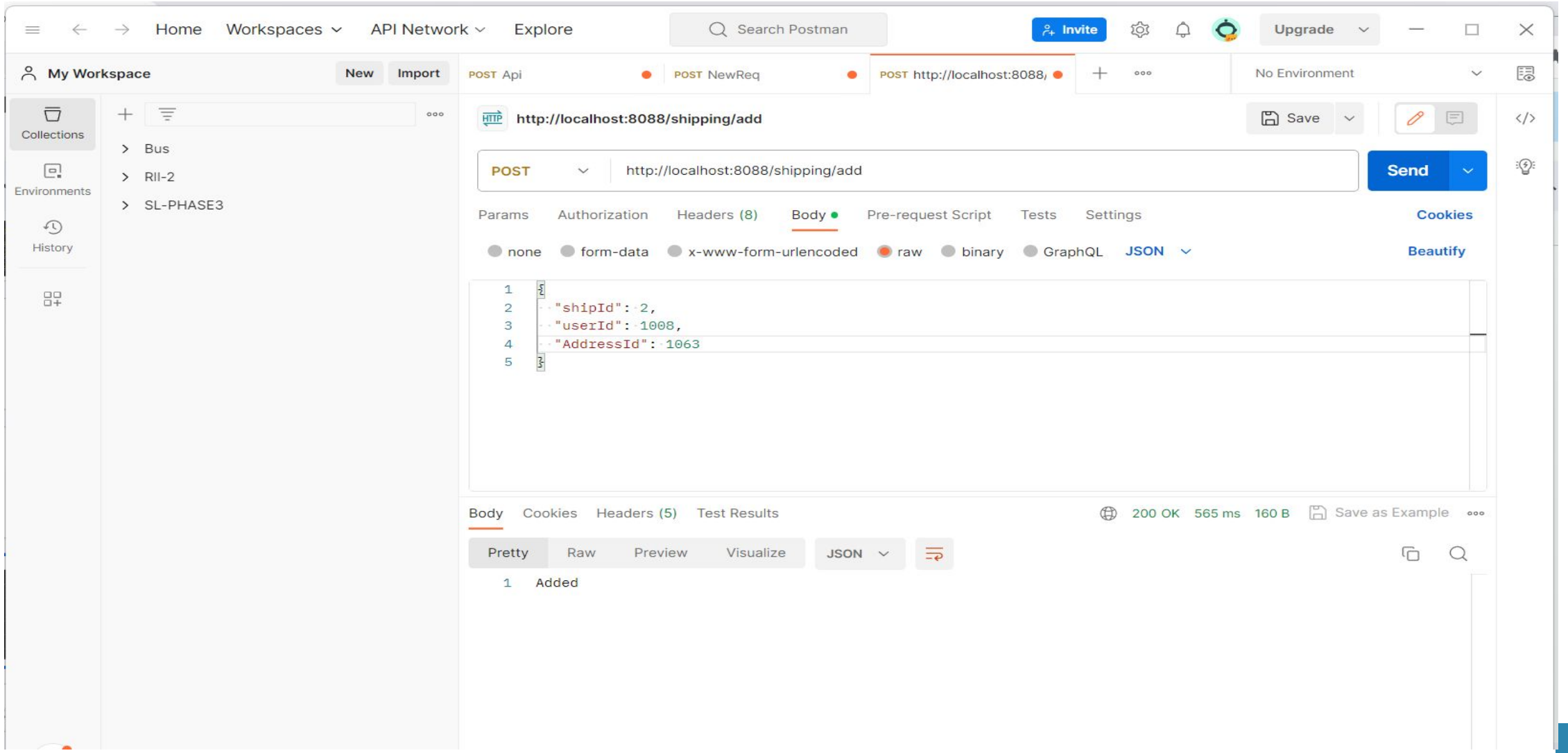
PAYMENT SERVICES



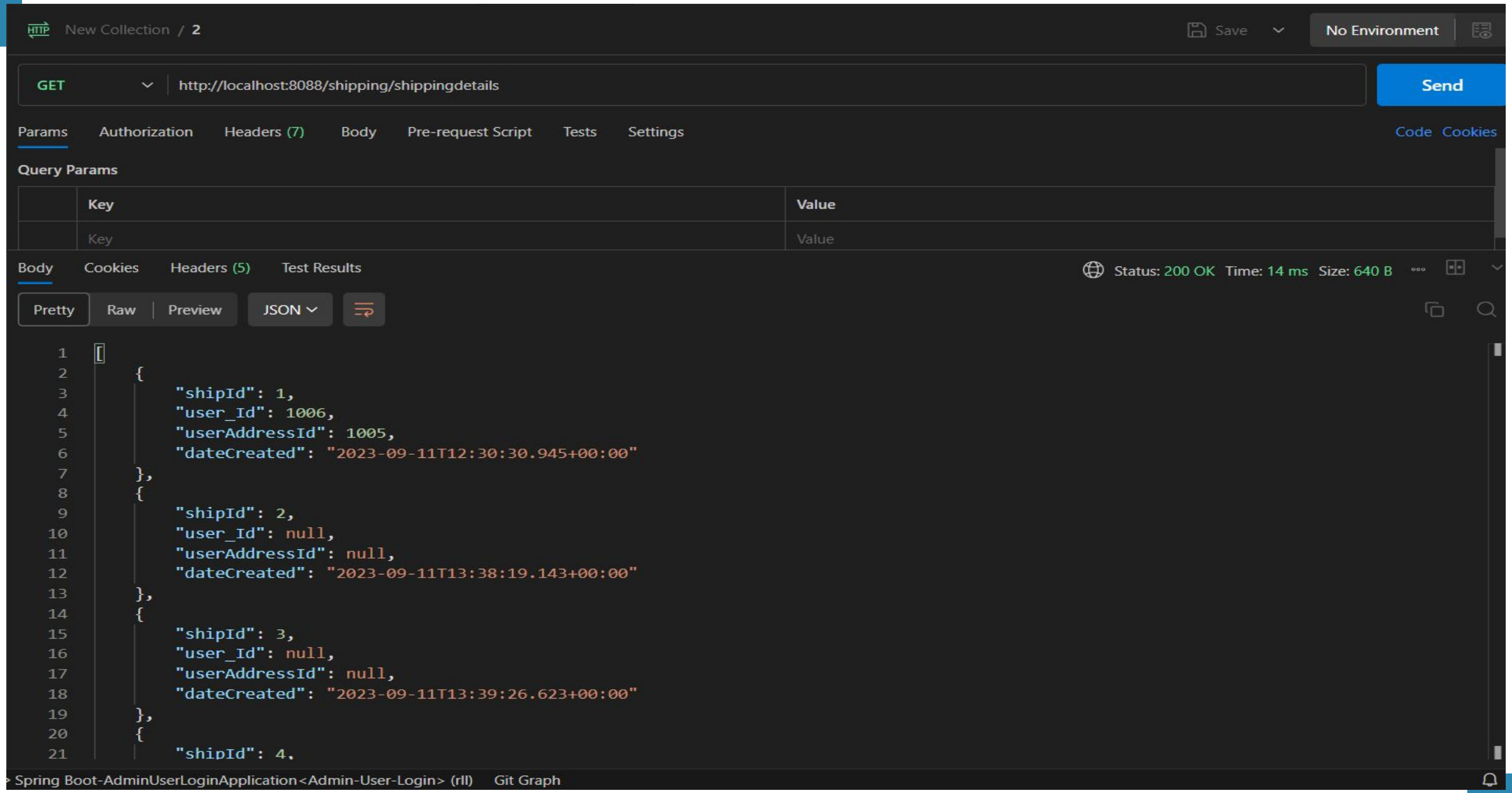
USER-SHIPPING SERVICE

Sinchana

- User shipping service will provide efficient and secure shipping options.



USER-SHIPPING SERVICE



CONCLUSION

- E-medicare microservices represent a transformative step towards modernizing and improving healthcare delivery.

- By enhancing efficiency, interoperability, patient engagement, and data security, they contribute to better user outcomes and cost-effective healthcare services.

- However, successful implementation requires careful planning, investment, and ongoing maintenance to ensure that the system operates effectively and securely.


THANK YOU