1.How To Test:

1. You can check the up ports with command “***netstat -an | find “LISTEN”***”

A screenshot of a computer screen

Description automatically generated

1. Start Eureka Server under (on port 8761)  
   “**D:\IdeaProjects\EurekaServerApp**”

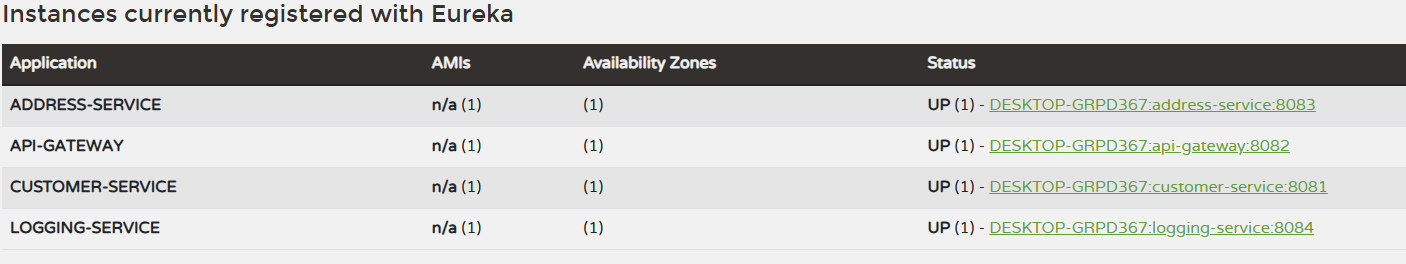
**java -jar target/eureka-server-0.0.1-SNAPSHOT.jar**

NOTE: if there is no related jar file in related path (/target/) create it first by “mvn clean install)

1. Start Keycloak (on port 8080)

**“D:\Keycloak\keycloak-26.0.7\keycloak-26.0.7\bin”  
kc.bat start-dev**

1. Start api-gateway under (on port 8082) “**D:\IdeaProjects\CWSMicroservicesUpd\api-gateway**”
2. Start address-service under (on port 8083)   
   “**D:\IdeaProjects\ CWSMicroservicesUpd \address-service**”
3. Start customer-service under (on port 8081)   
   “**D:\IdeaProjects\ CWSMicroservicesUpd \customer-service**”
4. Start logging-service under (on port 8084) (if working)  
   “**D:\IdeaProjects\ CWSMicroservicesUpd \logging-service**”
5. Check Eureka Server if you have related microservices. Open a browser, go to <http://localhost:8761/>. See related instances on the page:



1. Open Postman. Now you are able to send below requests:

|  |
| --- |
| IMPORTANT NOTE: For all of client requests, whether to address-service or customer-service, from now on, the customer web service requests only be sent over api-gateway, which has URL: **http://localhost:8082** |

1.1.Controls in Eureka and Keycloak:

If there is any problem occurs due to Eureka or Keycloak, to test both servers:

1. Eureka:

Check Eureka Server if you have related microservices. Open a browser, go to http://localhost:8761/. See related instances on the page:

A screenshot of a computer

Description automatically generated

1. Keycloak:

On any browser try to go [**http://localhost:8080/realms/customer-app-realm/.well-known/openid-configuration**](http://localhost:8080/realms/customer-app-realm/.well-known/openid-configuration)

If the page loads successfully and return JSON, then Keycloak is running properly:

A screenshot of a computer screen

Description automatically generated

If the page does not load, then Keycloak may not be running correctly. Restart Keycloak and try again.

# 2.Get JWT Token from Keycloak:

To access API Gateway, the client need to get a JWT token.

## 2.1.Request Token Using Postman:

1. Open **Postman**.
2. Create a **POST request** to:

|  |
| --- |
| http://localhost:8080/realms/customer-app-realm/protocol/openid-connect/token |

1. **Go to "Body" → "x-www-form-urlencoded"** and add:
   * grant\_type: password
   * client\_id: customer-service-client
   * client\_secret: IQMO1LDePpY3VtpA3dRBktD4UKNCXTqF
   * username: testuser
   * password: testuser1

A screenshot of a computer

Description automatically generated

1. Click **Send**.
   * If successful, you get a JWT token (access\_token).

|  |
| --- |
| {      "**access\_token**": "eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICI5NFRWWjlyeFdFWi14dEtwemVtaktUWFd4TERjLTNfOE9VQ2lpd051YWhBIn0..XqHAdrTbAnNibaNg9QriIAqa280V-Ogh6oZIJeIVhUCtz51mKfUpzDaT8i2YndjbWXiatm7A6gv8OxQnrmj0yZimPYwrAX24OzR2oi6YCzGD4UoGWcE93sUHFHozpA1pCGMyqDCuvKIUBqIZObzd-57QT0p65CdBa\_fb\_GapbVzda0SQtDvcElYcrxkcGVl6rlz53TJnc12T3YM51hDlGw\_-OOPVJfq\_DJDFMcgn6OmoKZpOixTmBwe5nfzlP1rVYoXthnG9Qhz5P3MHU76\_CuS5058RXX1bdCQIkvFJf04ORHBY-b6MjAk8sd1sBw5CvEUrITlRZB9b3Vw0FXRuNQ",      "**expires\_in**": 180,      "**refresh\_expires\_in**": 420,      "**refresh\_token**": "eyJhbGciOiJIUzUxMiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJjNTVlYjA4OS1lMGMwLTRmZTEtYTkzMy0xYjliM2U1MWM5MDQifQ..sjZoXrN73-vQ8w4WudXrdBbRvC-5jqlOFDz7jaagAvtPMevsVVDVTrKJOqCp4ty5P7AGs5aLD-HNFbvczqSjuw",      "**token\_type**": "Bearer",      "**not-before-policy**": 0,      "**session\_state**": "70ff0cc0-c9a8-4f1c-a67e-8556b351e9bc",      "**scope**": "profile email"  } |

* + **Copy the token** for the next step.

# 3.Test Without a Token Case:

These tests without token should fail:

|  |  |
| --- | --- |
| GET | http://localhost:8082/customers |
| Response | 401 Unauthorized |

# 4.Tests With JWT Token:

## 4.1.Get All Customers (GET):

|  |  |
| --- | --- |
| **GET** | http://localhost:8082/customers |
| **Header** | Authorization: Bearer {your-access-token} |
| **Response** | [      {          "id": 1,          "name": "Kaptan Mağara Adamı ",          "email": "kaptan.magara.adami@deneme.com",          "phoneNumber": "1234567890",          "addresses": [              {                  "id": 3,                  "addressName": "Work Foolish",                  "street": "Handan Sokak",                  "city": "İstanbul",                  "state": "Türkiye",                  "zipCode": "10001",                  "customerId": 1              }          ]      },      {          "id": 2,          "name": "İdil Biret",          "email": "idilbiret@gmail.com",          "phoneNumber": "05545909896",          "addresses": []      },      {          "id": 3,          "name": "Seda Çelebi",          "email": "sedaclbi@gmail.com",          "phoneNumber": "05545909896",          "addresses": [              {                  "id": 4,                  "addressName": "Home",                  "street": "Zuhal Sokak",                  "city": "İstanbul",                  "state": "Türkiye",                  "zipCode": "356660",                  "customerId": 3              },              {                  "id": 5,                  "addressName": "İş",                  "street": "Zuhaliş Sokak",                  "city": "İstanbul",                  "state": "Türkiye",                  "zipCode": "356660",                  "customerId": 3              },              {                  "id": 6,                  "addressName": "Home Music",                  "street": "Muzisyes Sokak",                  "city": "İstanbul",                  "state": "Türkiye",                  "zipCode": "10001",                  "customerId": 3              }          ]      },      {          "id": 11,          "name": "Roei Morgen ghgdf1",          "email": "roei.morgen.ghdfg1@deneme.com",          "phoneNumber": "1234567890",          "addresses": [              {                  "id": 7,                  "addressName": "Work",                  "street": "Handan Sokak",                  "city": "İstanbul",                  "state": "Türkiye",                  "zipCode": "10001",                  "customerId": 11              }          ]      }  ] |

# 3.Get Customer by ID (GET):

|  |  |
| --- | --- |
| **GET** | [http://localhost:8082/customers/{customerId}](http://localhost:8082/customers/%7bcustomerId%7d)  http://localhost:8082/customers/3 |
| **Response** | {      "id": 3,      "name": "Seda Çelebi",      "email": "sedaclbi@gmail.com",      "phoneNumber": "05545909896",      "addresses": [          {              "id": 4,              "addressName": "Home",              "street": "Zuhal Sokak",              "city": "İstanbul",              "state": "Türkiye",              "zipCode": "356660",              "customerId": 3          },          {              "id": 5,              "addressName": "İş",              "street": "Zuhaliş Sokak",              "city": "İstanbul",              "state": "Türkiye",              "zipCode": "356660",              "customerId": 3          },          {              "id": 6,              "addressName": "Home Music",              "street": "Muzisyes Sokak",              "city": "İstanbul",              "state": "Türkiye",              "zipCode": "10001",              "customerId": 3          }      ]  } |

If the customerId is not valid in DB, it returns “**404 – Not Found:Customer not found!**”

# 4.Create a Customer (POST):

Without address

|  |  |
| --- | --- |
| **POST** | http://localhost:8082/customers |
| **Headers** | Content-Type: application/json |
| **Body** | {      "name": "Freddy Krugger",      "email": "freddy.krugger@deneme.com",      "phoneNumber": "1234567890",      "addresses": [          {              "addressName": "Work",              "street": "Elm Street",              "city": "New York",              "state": "USD",              "zipCode": "10001"          }      ]  } |
| **Response** | **201 - Created** {      "id": 12,      "name": "Freddy Krugger",      "email": "freddy.krugger@deneme.com",      "phoneNumber": "1234567890",      "addresses": [          {              "id": **null**,              "addressName": "Work",              "street": "Elm Street",              "city": "New York",              "state": "USD",              "zipCode": "10001",              "customerId": 12          }      ]  } |
| **POST** | http://localhost:8082/customers |
| **Headers** | Content-Type: application/json |
| **Body** | {      "name": "Burhan Kuzu",      "email": "burhan.kuzu@deneme.com",      "phoneNumber": "1234567890"  } |
| **Response** | **201 - Created** {      "id": 13,      "name": "Burhan Kuzu",      "email": "burhan.kuzu@deneme.com",      "phoneNumber": "1234567890",      "addresses": **null**  } |

# 5.Delete Customer (DELETE):

|  |  |
| --- | --- |
| **DELETE** | http://localhost:8082/customers/{customerId} |
| **Response** | {  "message": "Customer deleted successfully"  } |
| **DELETE** | http://localhost:8082/customers/58 |
| **Response** | 200 – OK: Customer and related addresses deleted successfully. |
| **DELETE** | http://localhost:8082/customers/61 |
| **Response** | 200 – OK: Customer and related addresses deleted successfully. |

If the URL contains a customerId that is not valid in customer table in DB, it will return “**404 – Not Found:Customer not found!**”

# 6.Get All Addresses (GET):

|  |  |
| --- | --- |
| **GET** | http://localhost:8082/addresses |
| **Response** | [      {          "id": 3,          "addressName": "Work Foolish",          "street": "Handan Sokak",          "city": "İstanbul",          "state": "Türkiye",          "zipCode": "10001",          "customerId": 1      },      {          "id": 4,          "addressName": "Home",          "street": "Zuhal Sokak",          "city": "İstanbul",          "state": "Türkiye",          "zipCode": "356660",          "customerId": 3      },      {          "id": 5,          "addressName": "İş",          "street": "Zuhaliş Sokak",          "city": "İstanbul",          "state": "Türkiye",          "zipCode": "356660",          "customerId": 3      },      {          "id": 6,          "addressName": "Home Music",          "street": "Muzisyes Sokak",          "city": "İstanbul",          "state": "Türkiye",          "zipCode": "10001",          "customerId": 3      },      {          "id": 7,          "addressName": "Work",          "street": "Handan Sokak",          "city": "İstanbul",          "state": "Türkiye",          "zipCode": "10001",          "customerId": 11      },      {          "id": 8,          "addressName": "Work",          "street": "Elm Street",          "city": "New York",          "state": "USD",          "zipCode": "10001",          "customerId": 12      }  ] |

# 7.Get All Addresses of a Customer (GET):

|  |  |
| --- | --- |
| **GET** | [http://localhost:8082/addresses/customers/{customerId}](http://localhost:8082/addresses/customers/%7bcustomerId%7d)  http://localhost:8082/addresses/customers/3 |
| **Response** | [      {          "id": 4,          "addressName": "Home",          "street": "Zuhal Sokak",          "city": "İstanbul",          "state": "Türkiye",          "zipCode": "356660",          "customerId": 3      },      {          "id": 5,          "addressName": "İş",          "street": "Zuhaliş Sokak",          "city": "İstanbul",          "state": "Türkiye",          "zipCode": "356660",          "customerId": 3      },      {          "id": 6,          "addressName": "Home Music",          "street": "Muzisyes Sokak",          "city": "İstanbul",          "state": "Türkiye",          "zipCode": "10001",          "customerId": 3      }  ] |

If the customerId is not valid in DB, it returns “**404 – Not Found:Customer not found!**”

# 8.Delete an Address of a Customer (DELETE):

|  |  |
| --- | --- |
| **DELETE** | http://localhost:8082/customers/{customerId} |
| **Response** | {  "message": "Customer deleted successfully"  } |
| **DELETE** | http://localhost:8082/customers/58 |
| **Response** | 200 – OK: Customer and related addresses deleted successfully. |
| **DELETE** | http://localhost:8082/customers/61 |
| **Response** | 200 – OK: Customer and related addresses deleted successfully. |

If the URL contains a customerId that is not valid in customer table in DB, it will return “**404 – Not Found:Customer not found!**”