# Credit Card Microservices Demo Project Guide

## System Requirements

1. **Docker Desktop**: Required to run Docker containers.
2. **Postman**: For testing the APIs by importing and running collections.

## Steps to Run the Project

1. **Download the Project:**

* Clone the project from GitHub using this link:

[**https://github.com/yoramnag/credit-card-microservices-demo.git**](https://github.com/yoramnag/credit-card-microservices-demo.git)

1. **Set Up Docker:**

* Open **Docker Desktop** and ensure the Docker engine is running.

1. **Navigate to Project Folder:**

* Open **PowerShell** or **IntelliJ IDEA** and navigate to the folder where the docker-compose.yml file is located.
* You can use the following PowerShell command to change the directory:

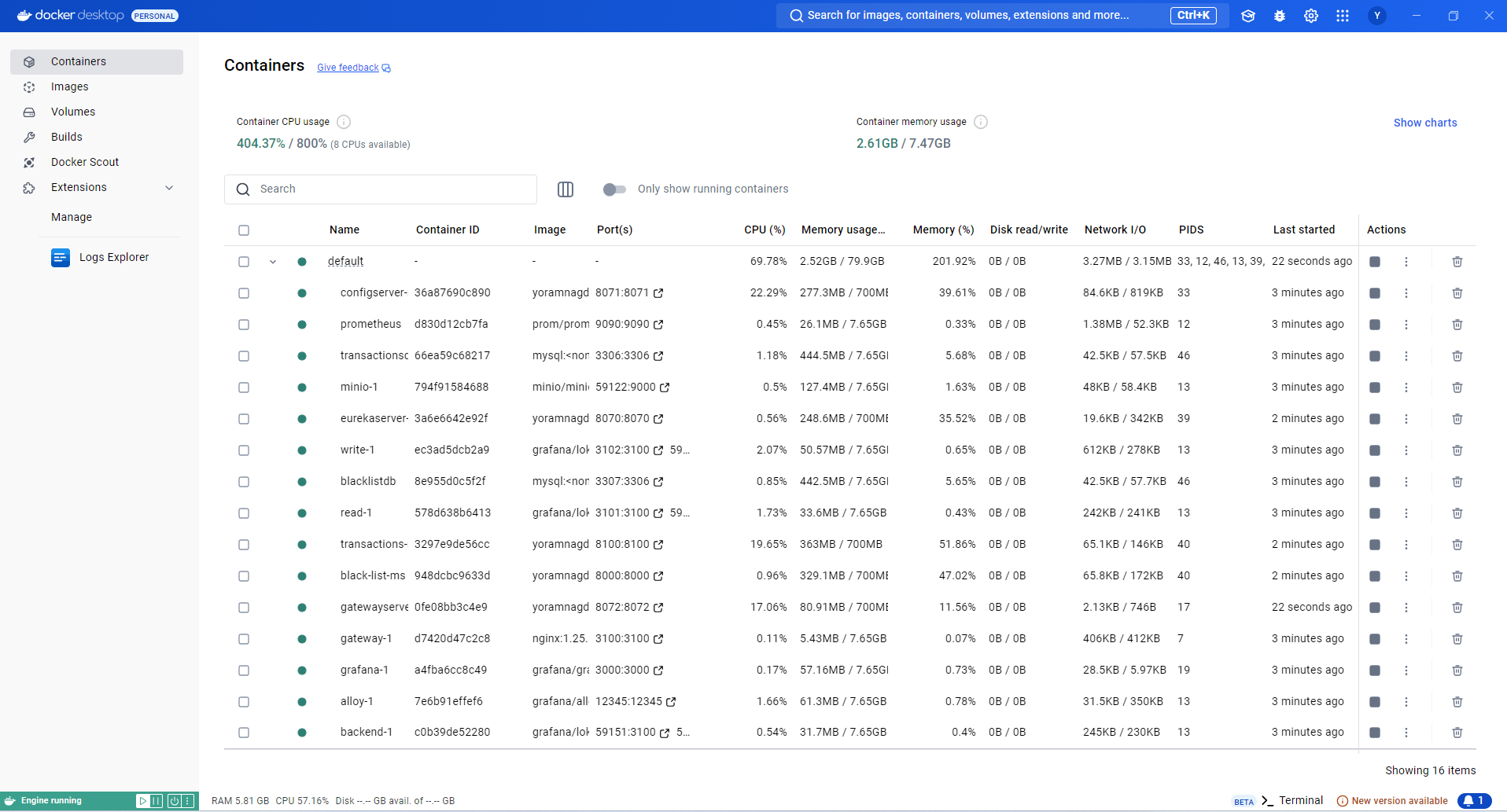
cd path\to\your\project

1. **Start Docker Containers:**

* In the project folder, run the following command to start the Docker containers:

docker compose up -d

* After the command executes successfully, go to **Docker Desktop** and verify that the necessary containers are running.

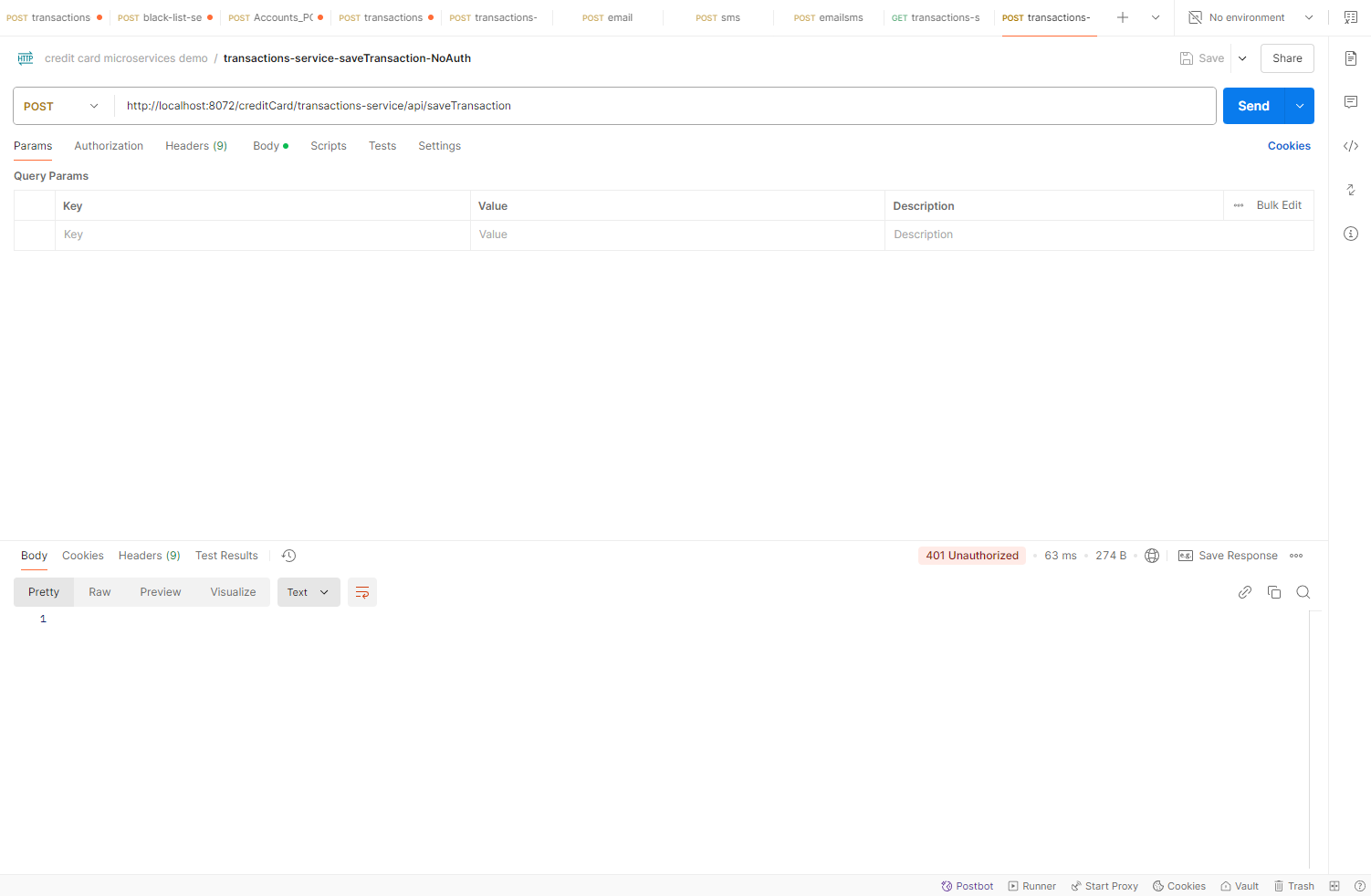


## Testing the APIs with Postman

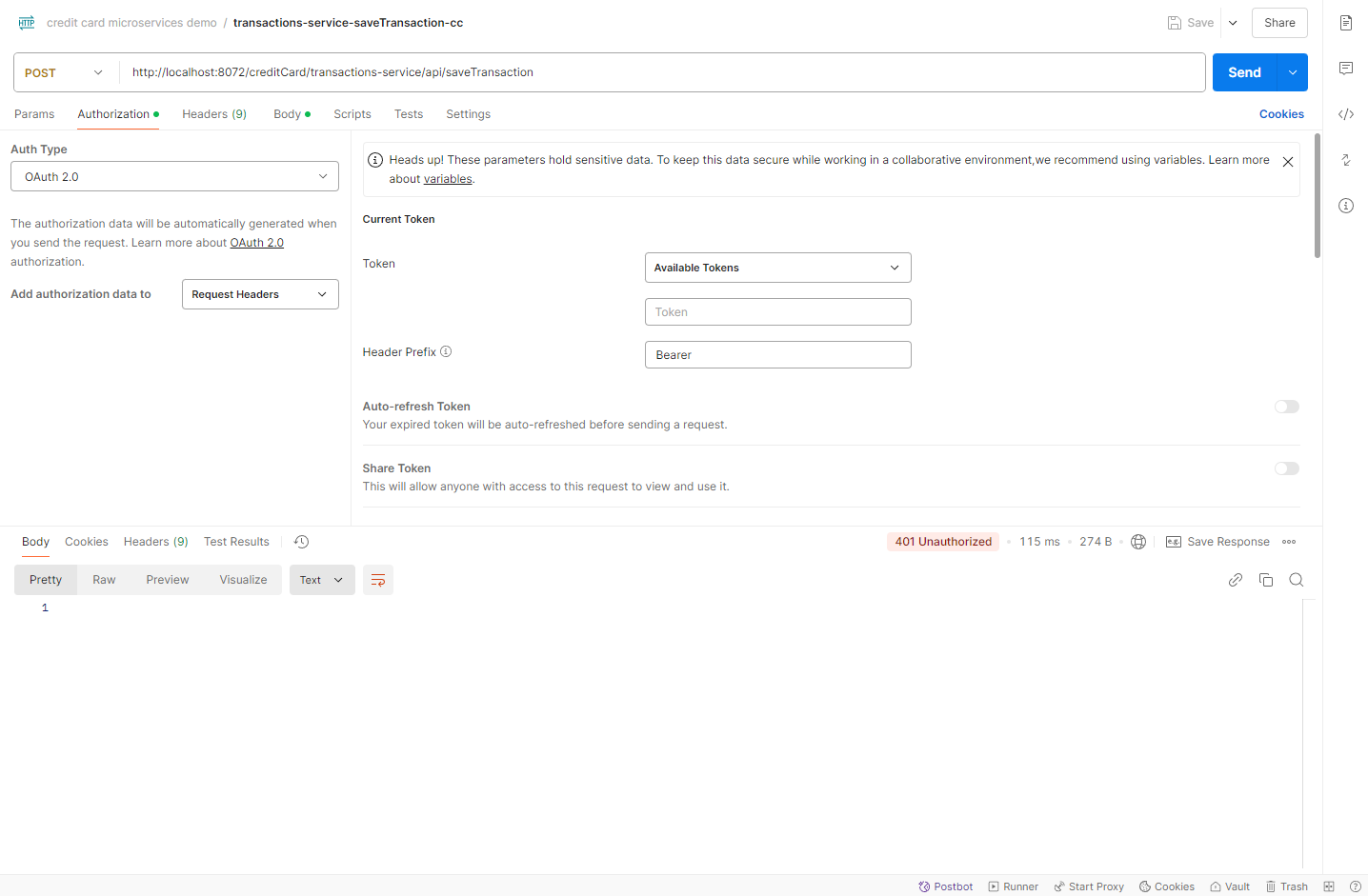
1. **Import the Postman Collection:**

* Open **Postman**.
* Go to **File > Import**, then select the credit card microservices demo.postman\_collection.json file from your project directory. This file contains all the API requests required to interact with your services.

1. **Run the Save Transaction Request:**

* In Postman, locate the request named **transactions-service-saveTransaction-NoAuth** from the imported collection.
* Run the request. You will receive a **401 Unauthorized** response. This is expected.

1. **Run the Request with Authorization:**

* Now, try to run the request named **transactions-service-saveTransaction-cc**. This request includes **OAuth 2.0 authorization**.
* You will again receive a **401 Unauthorized** response because the client-id creditCard-callcenter-cc doesn’t exist in Keycloak.

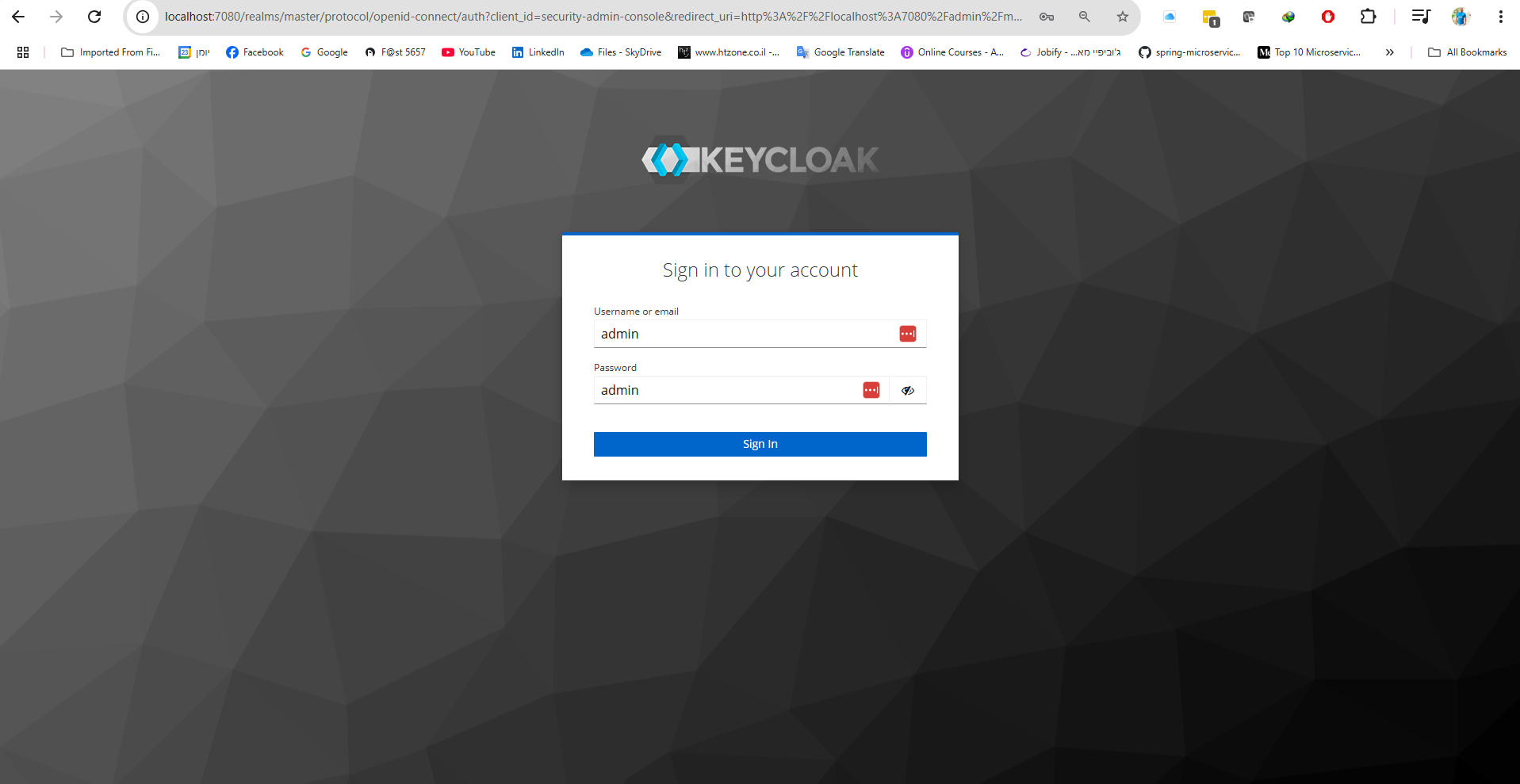
## Set Up Client in Keycloak

1. **Access Keycloak Admin Console:**

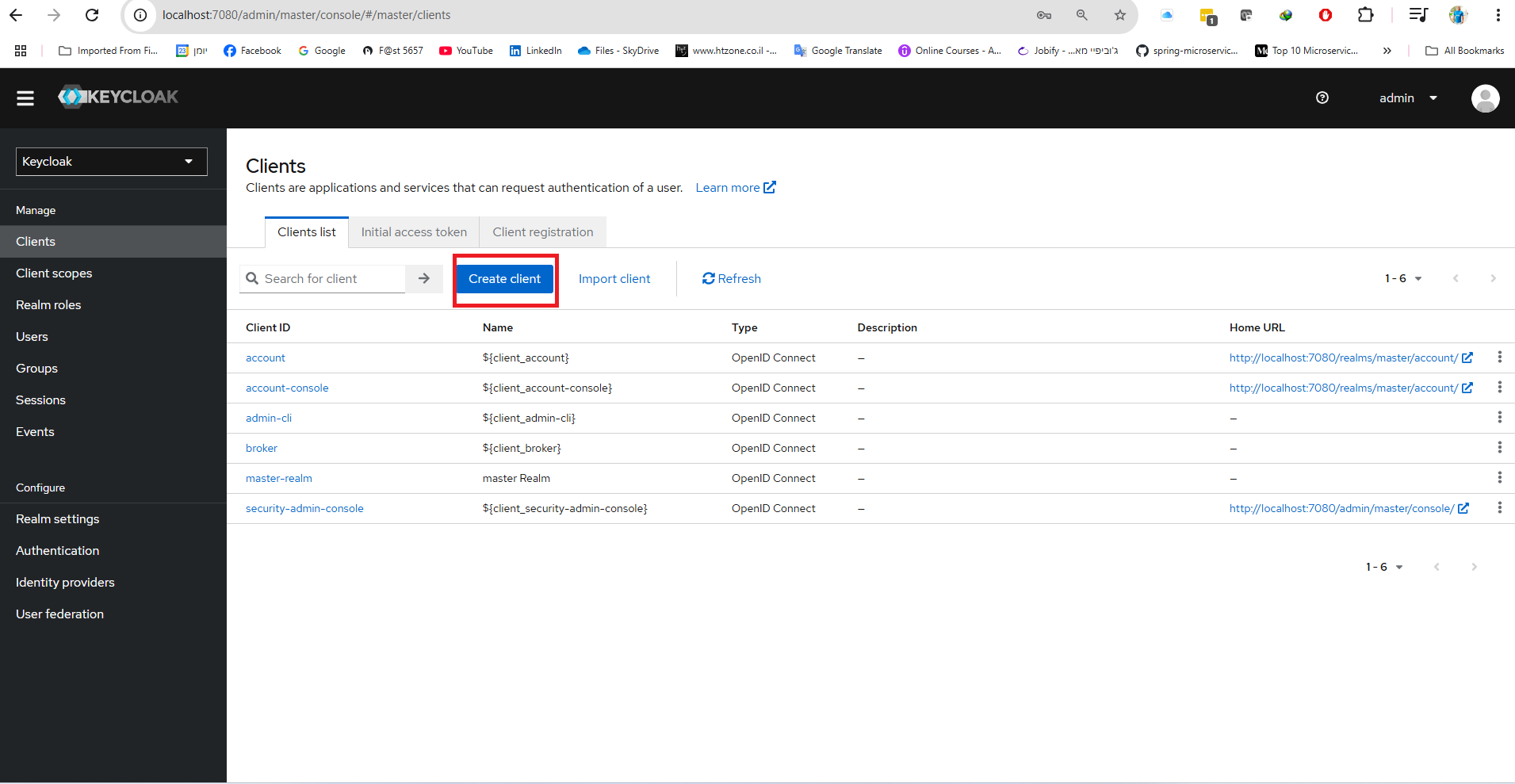
Go to: <http://localhost:7080/admin/master/console/#/master/clients>

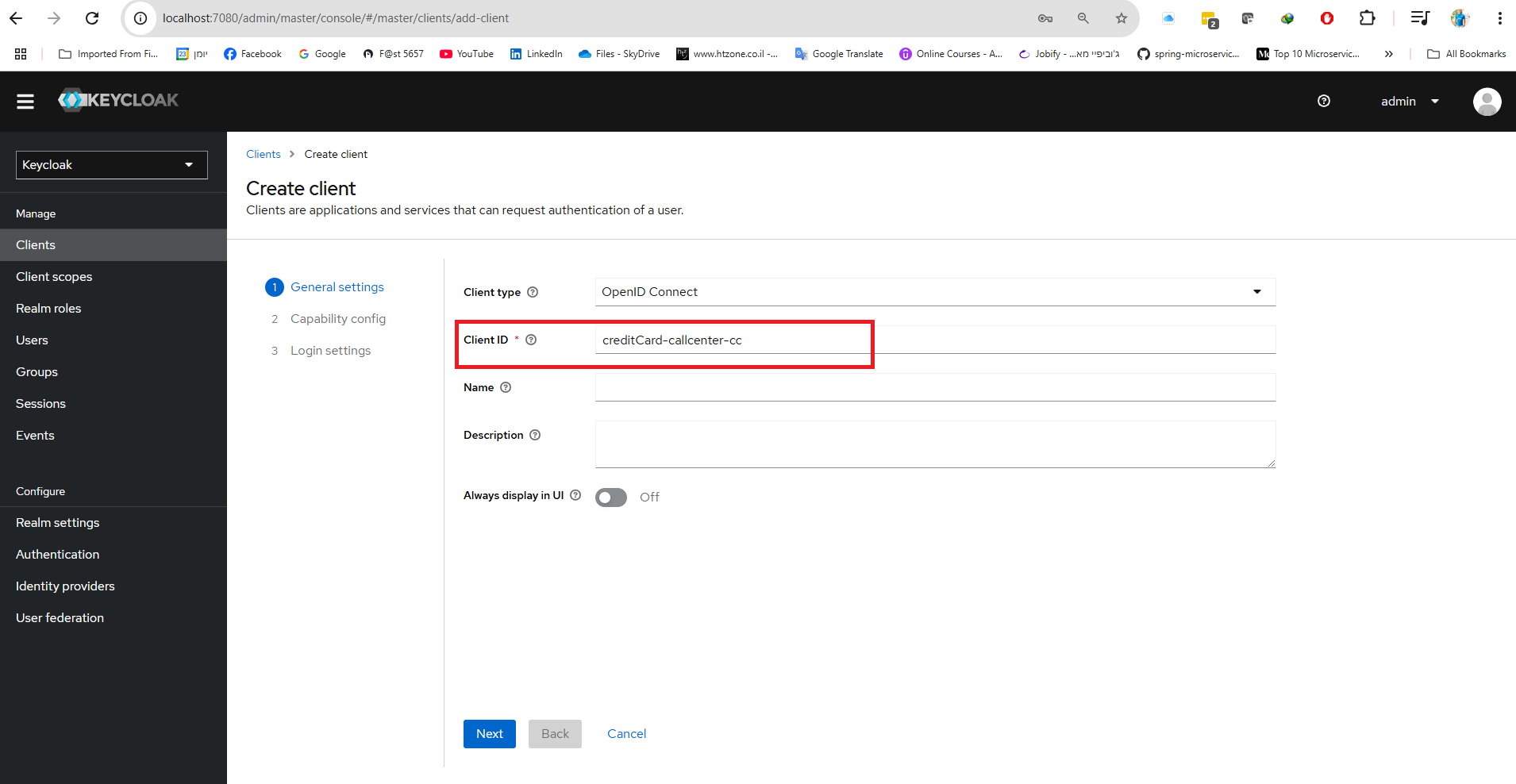
Login with the following credentials:

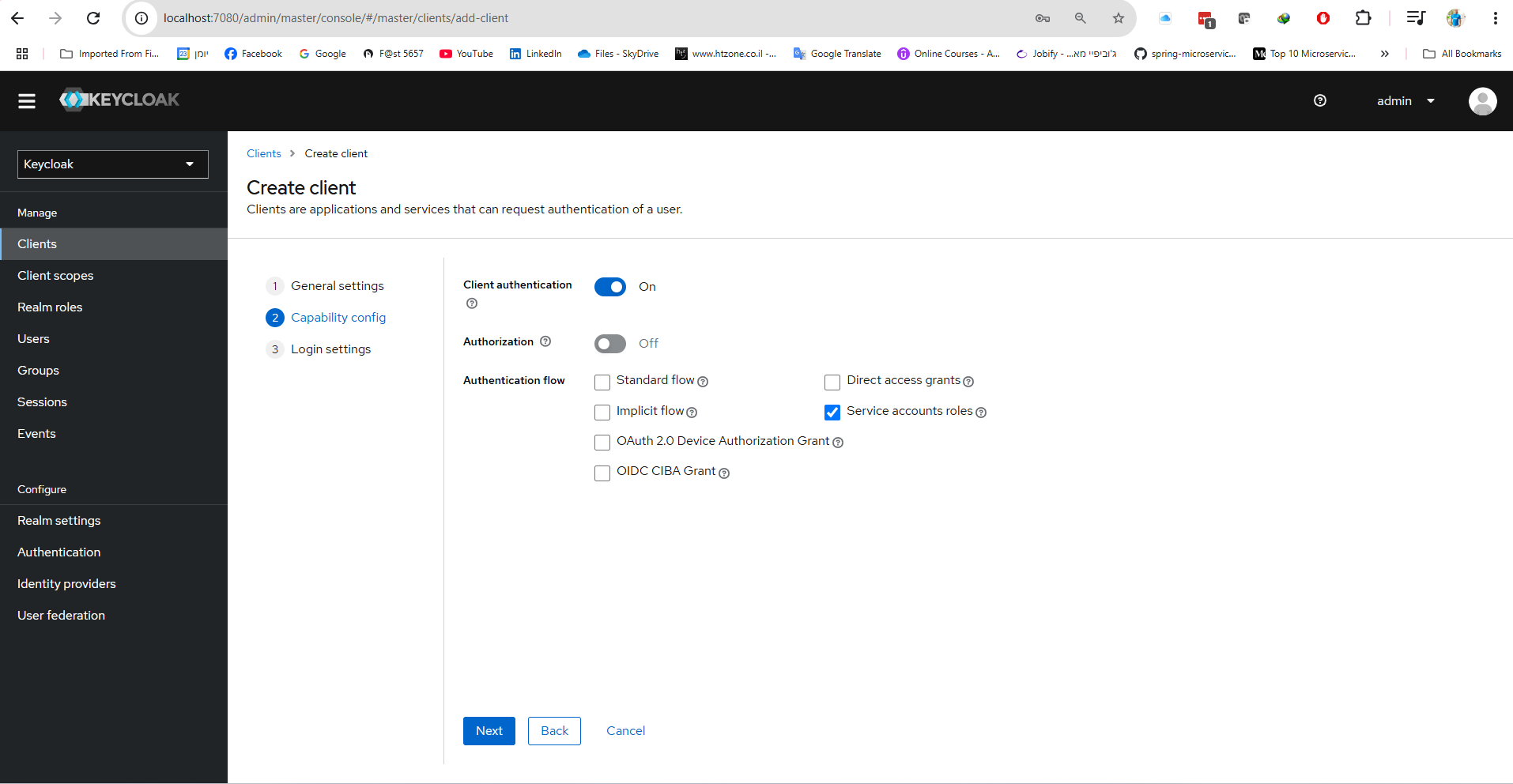
**Username**: admin

**Password**: admin

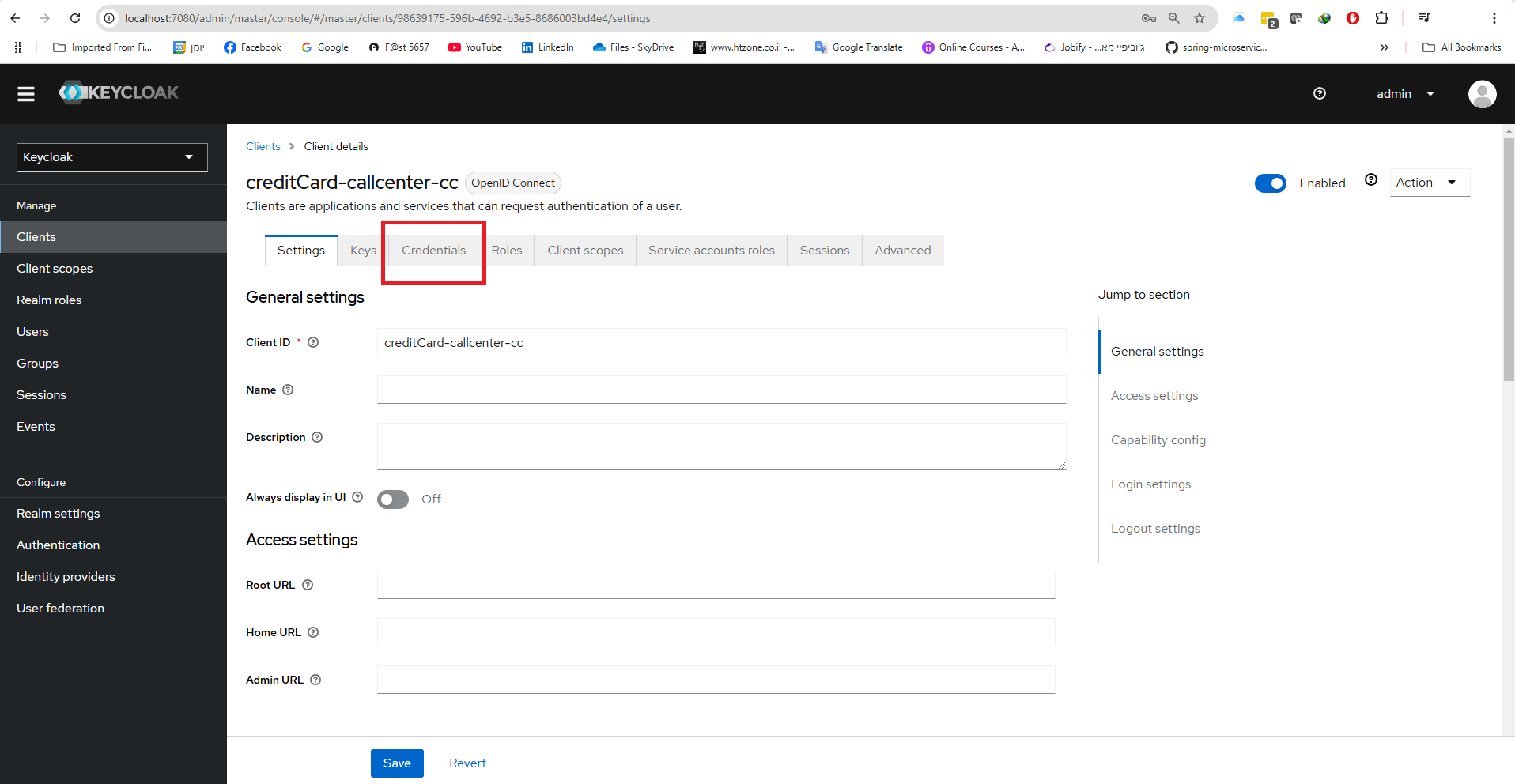
1. **Create a New Client:**

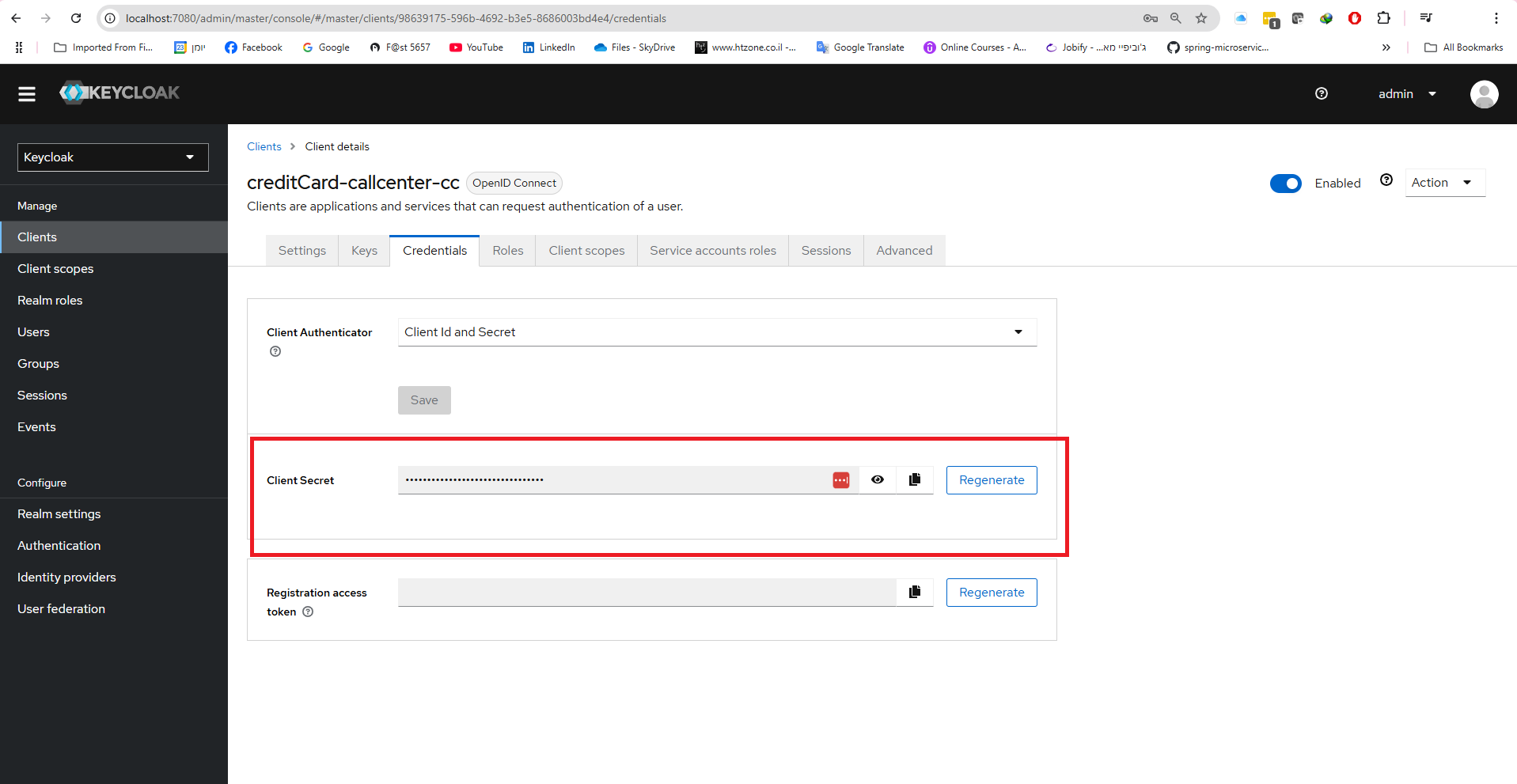
* In the **left sidebar**, select **Clients** and then click on **Create Client**.
* Enter creditCard-callcenter-cc as the **Client ID** and click **Next**.
* Enable **Client Authentication** and check only **Service Accounts Roles**.
* **Click Next.**



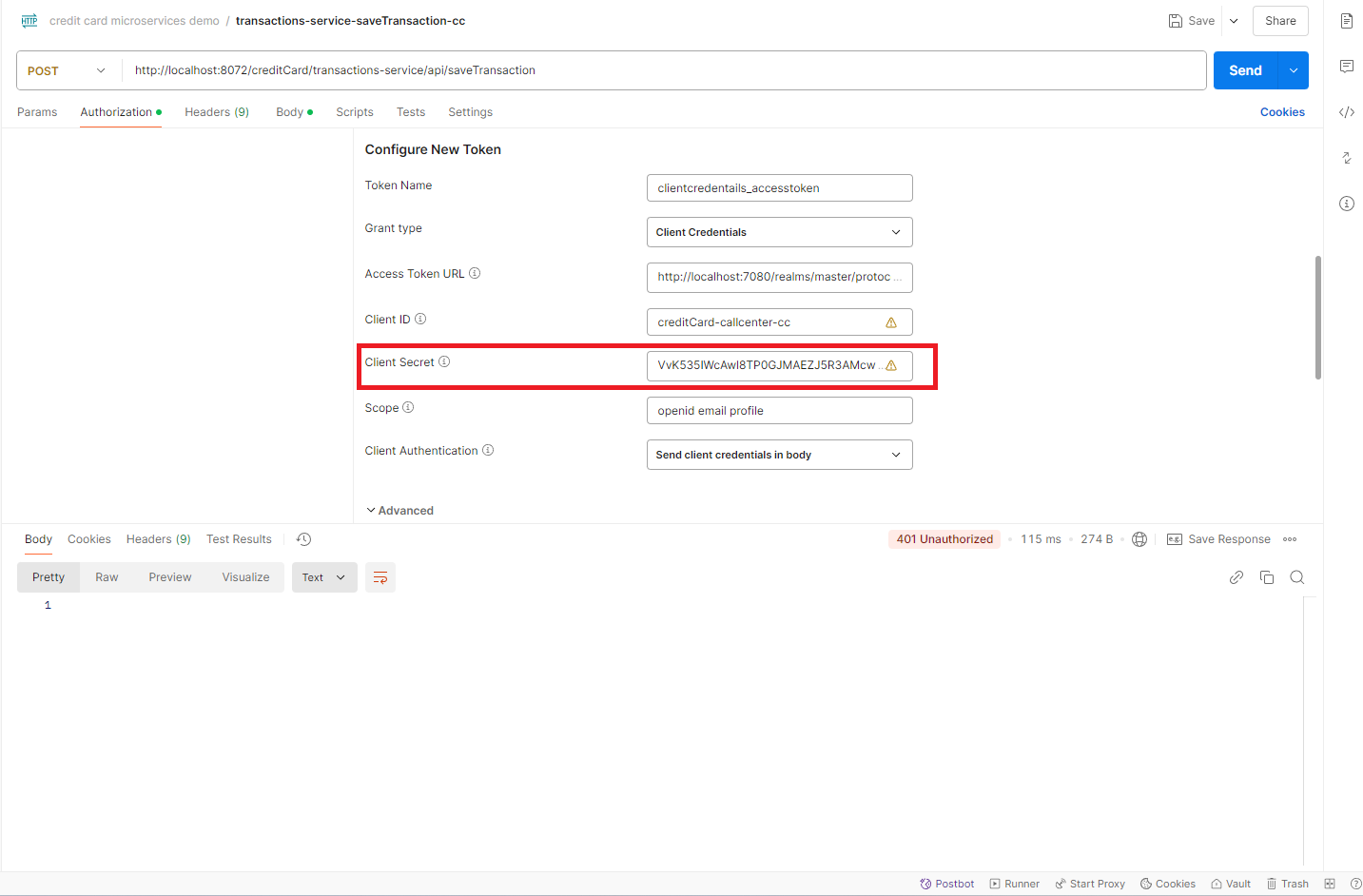


1. **Generate Client Credentials:**

* Click on **Credentials**.
* **Copy the Client Secret** and save it.



1. **Update Postman with the Client Secret:**

* Go back to Postman.
* In the **transactions-service-saveTransaction-cc** request, update the **Authorization** header with the **client secret** you just copied.

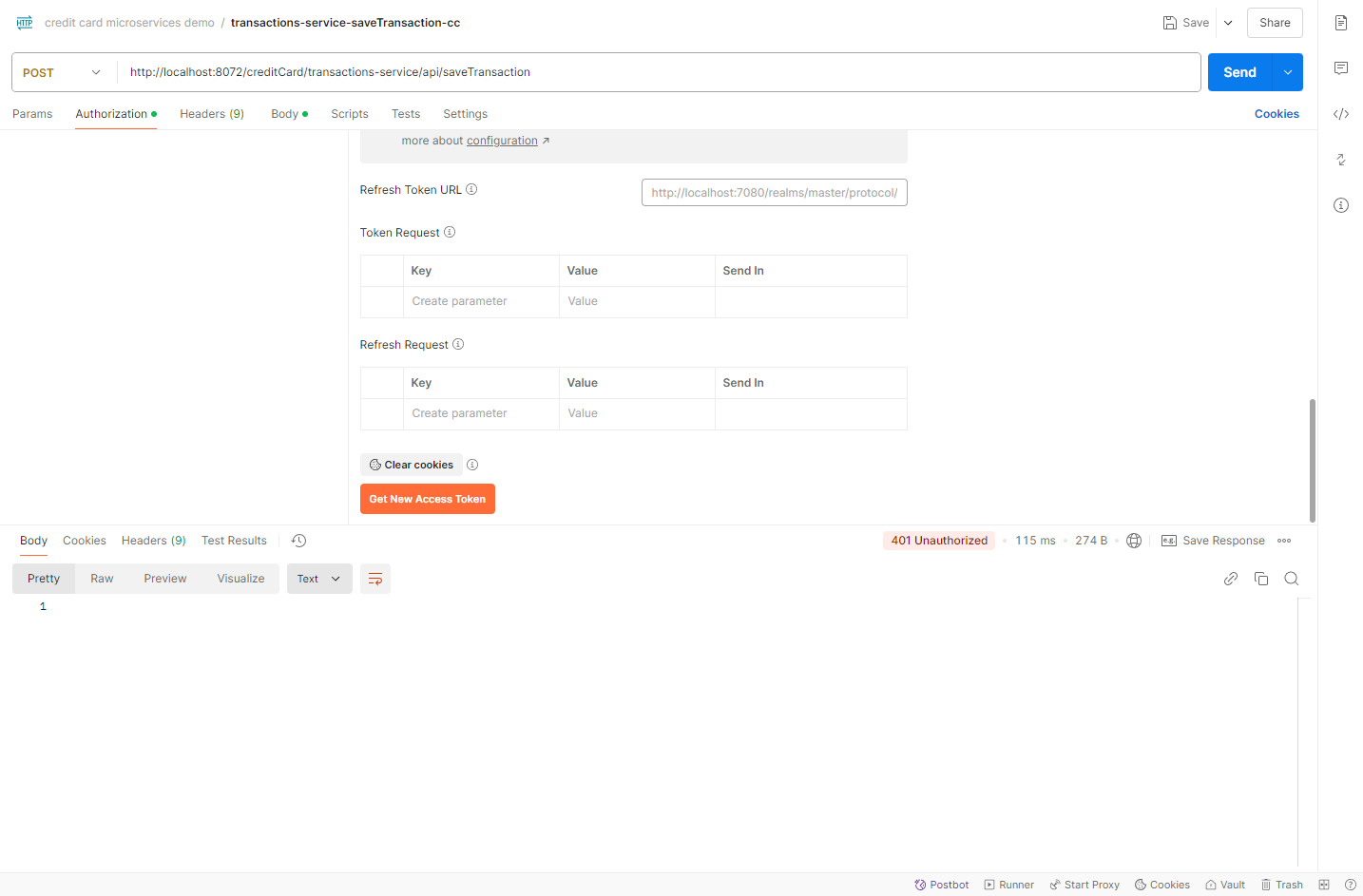
## Obtain Access Token in Postman

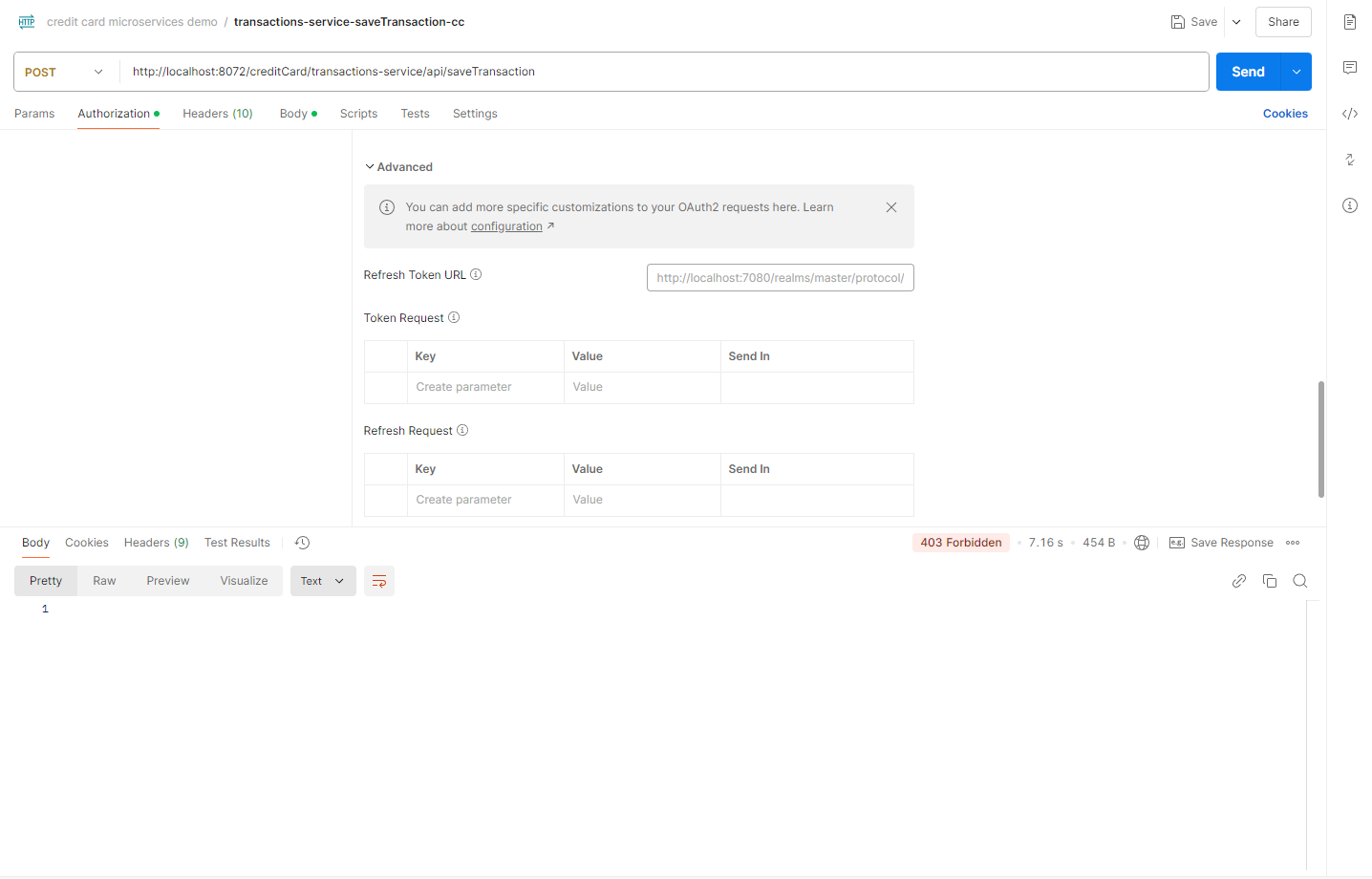
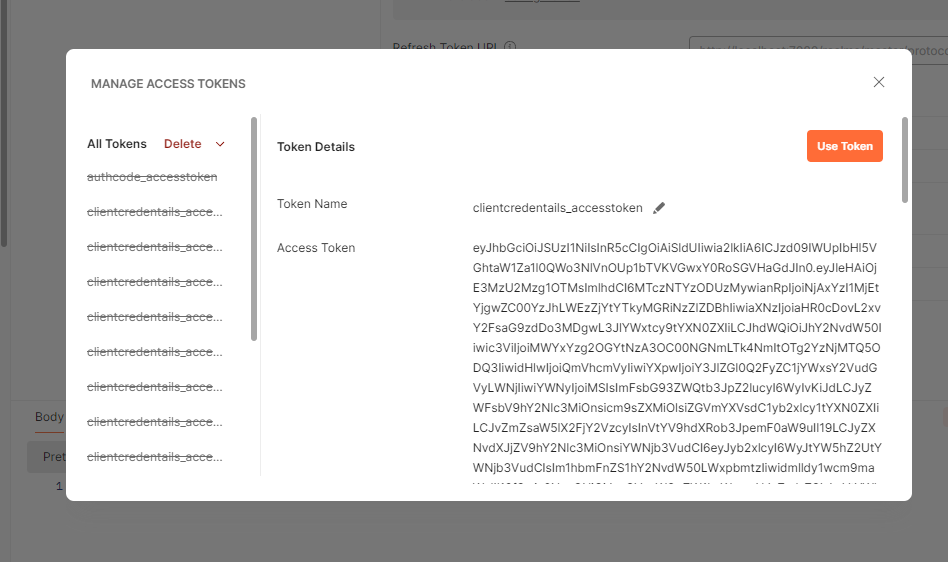
1. **Get New Access Token:**

* In Postman, click **Get New Access Token**.
* Once the token is generated, click **Use Token**.

1. **Run the Request Again:**

* Try running the transactions-service-saveTransaction-cc request again. You should receive a **403 Forbidden** response.



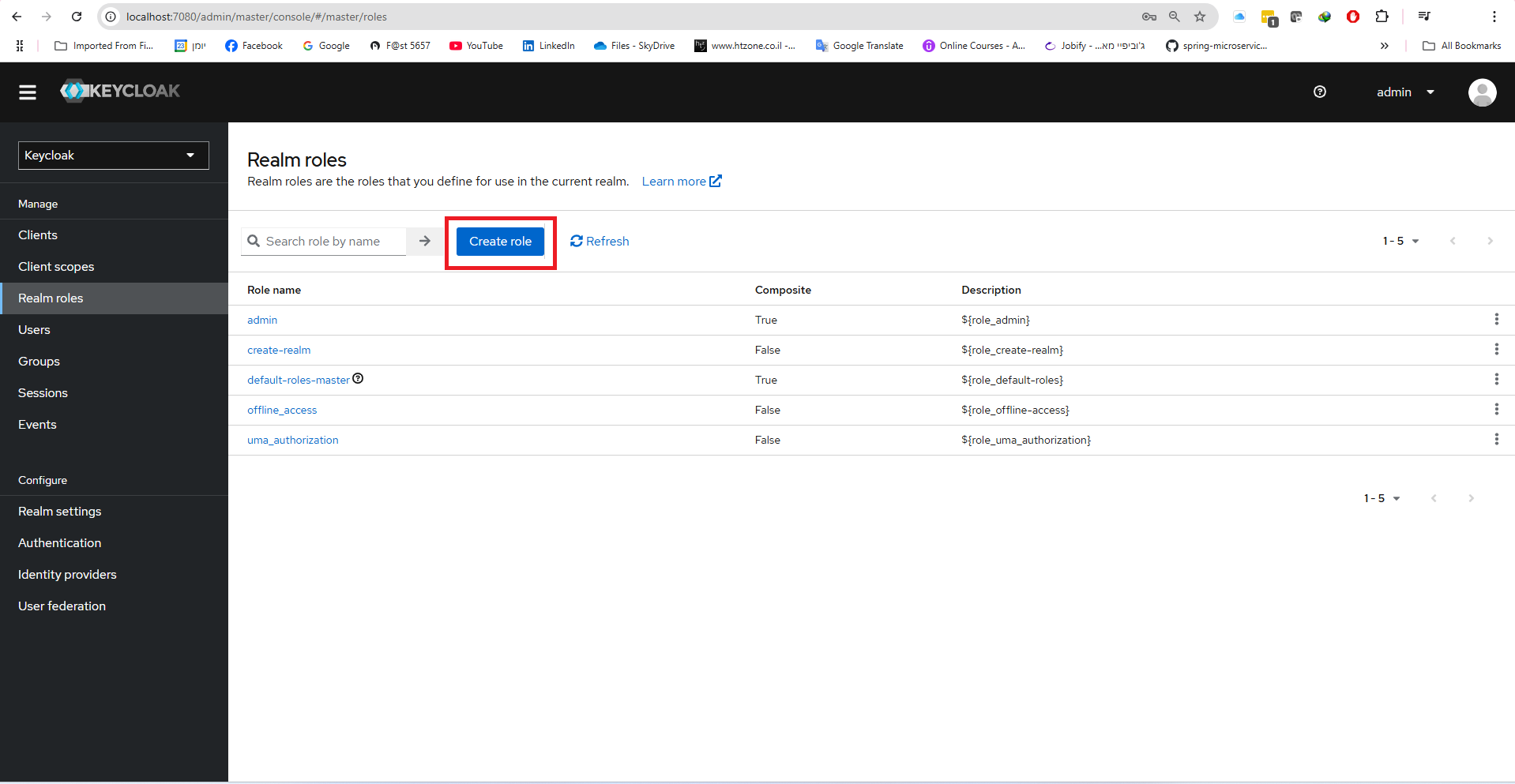


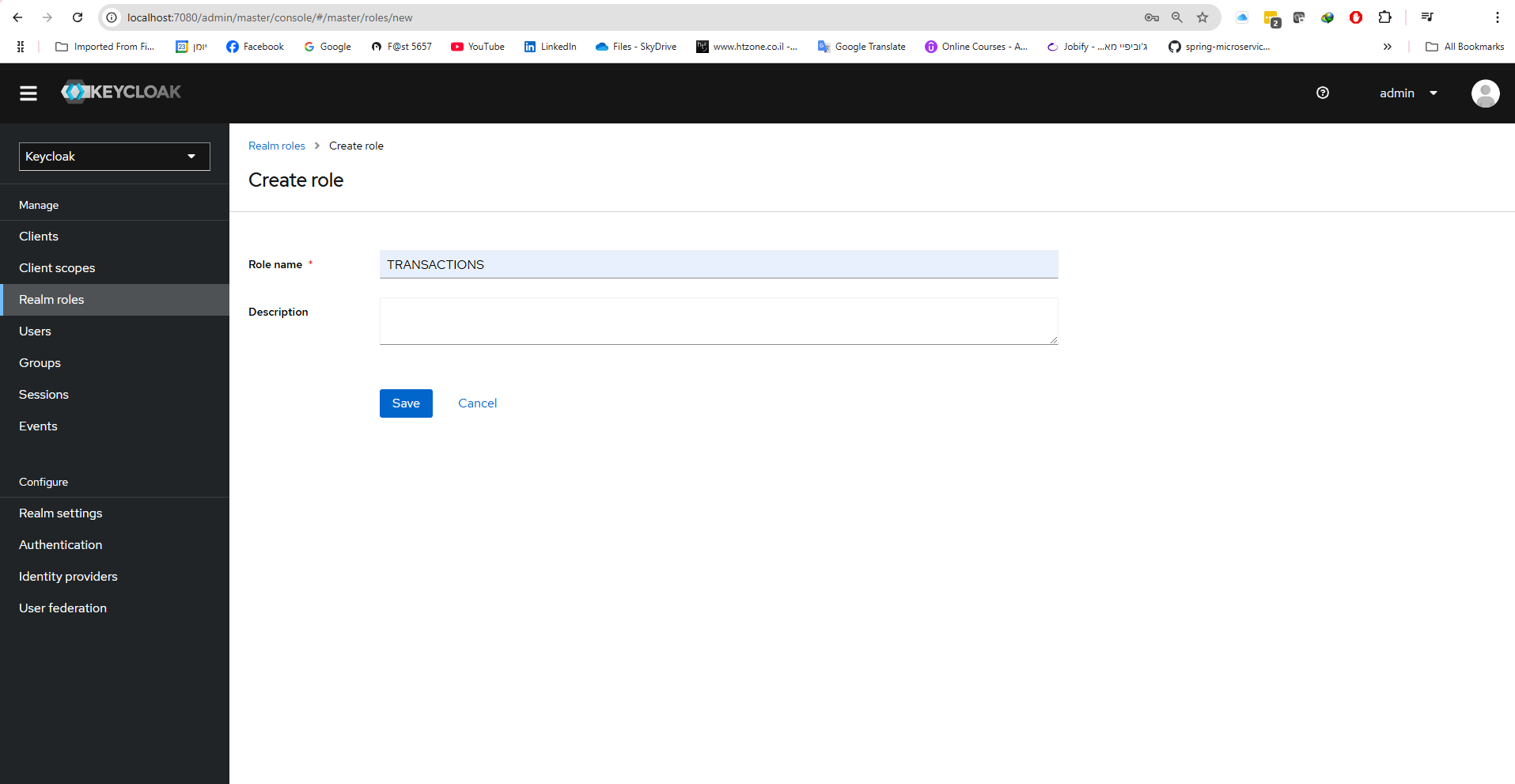
## Assign Roles to Client in Keycloak

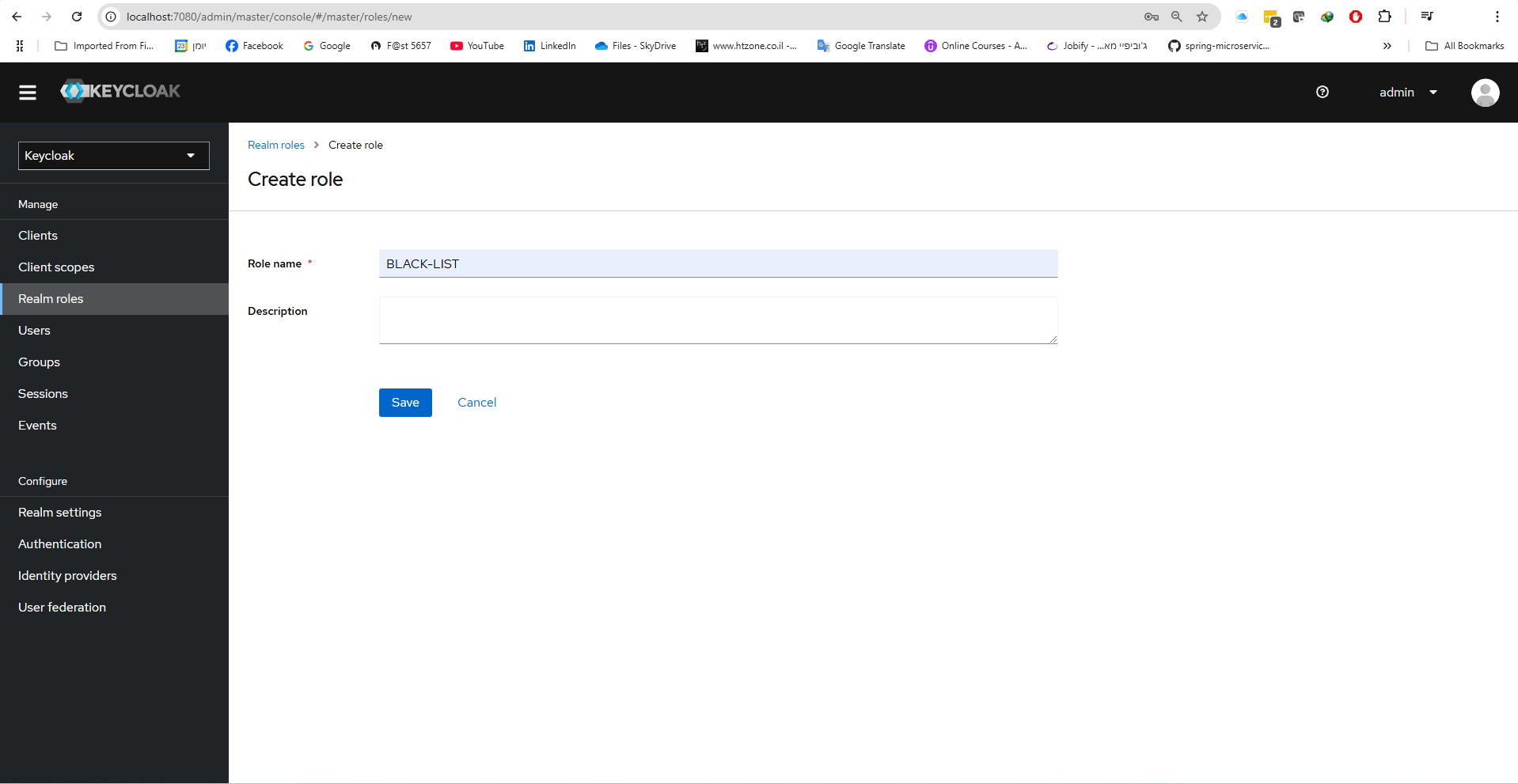
1. **Create New Roles in Keycloak:**

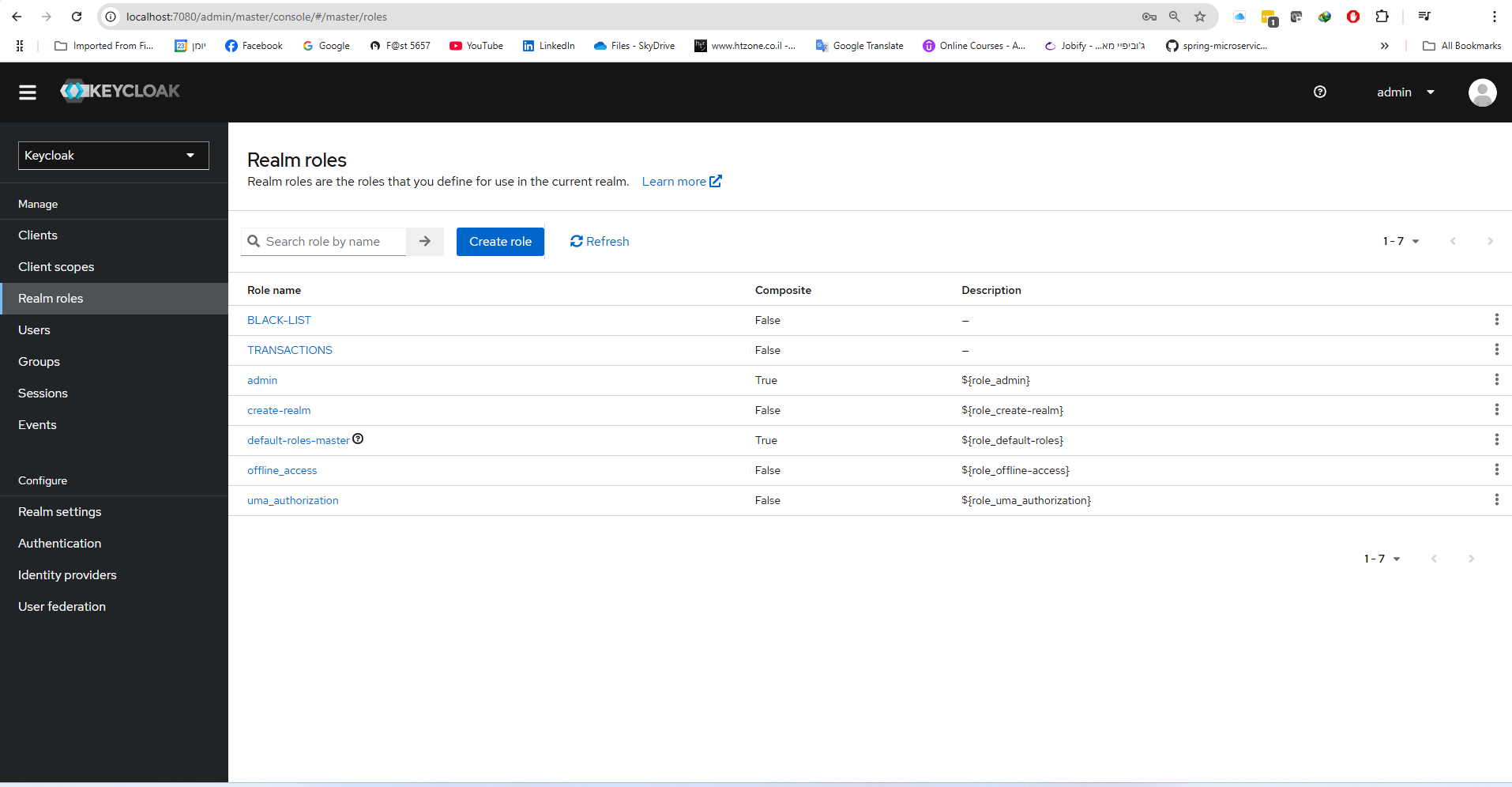
* In Keycloak, click **Realm Roles** in the left sidebar.
* Click **Create Role** and create the following roles (uppercase required):

TRANSACTIONS

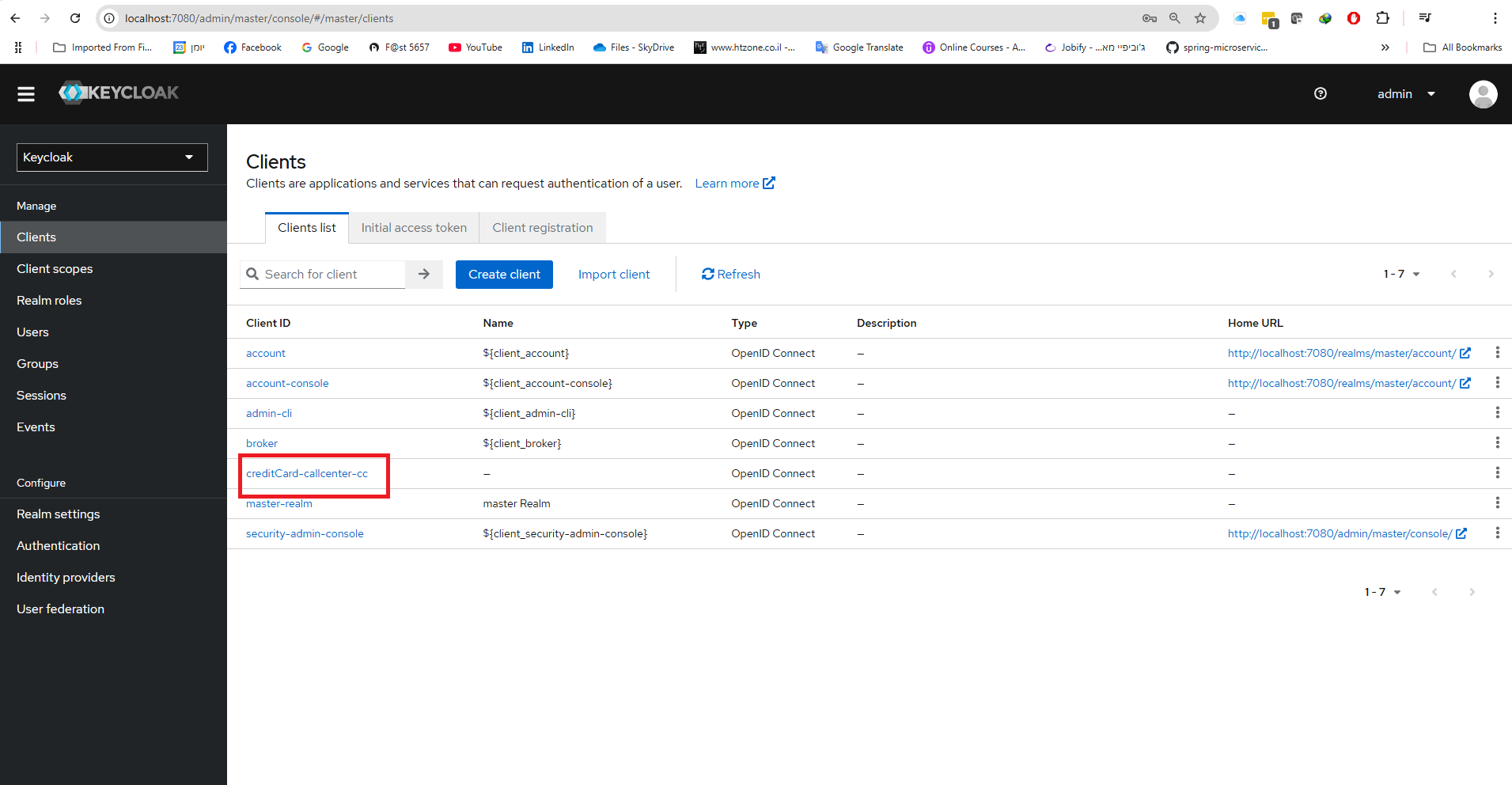
BLACK-LIST

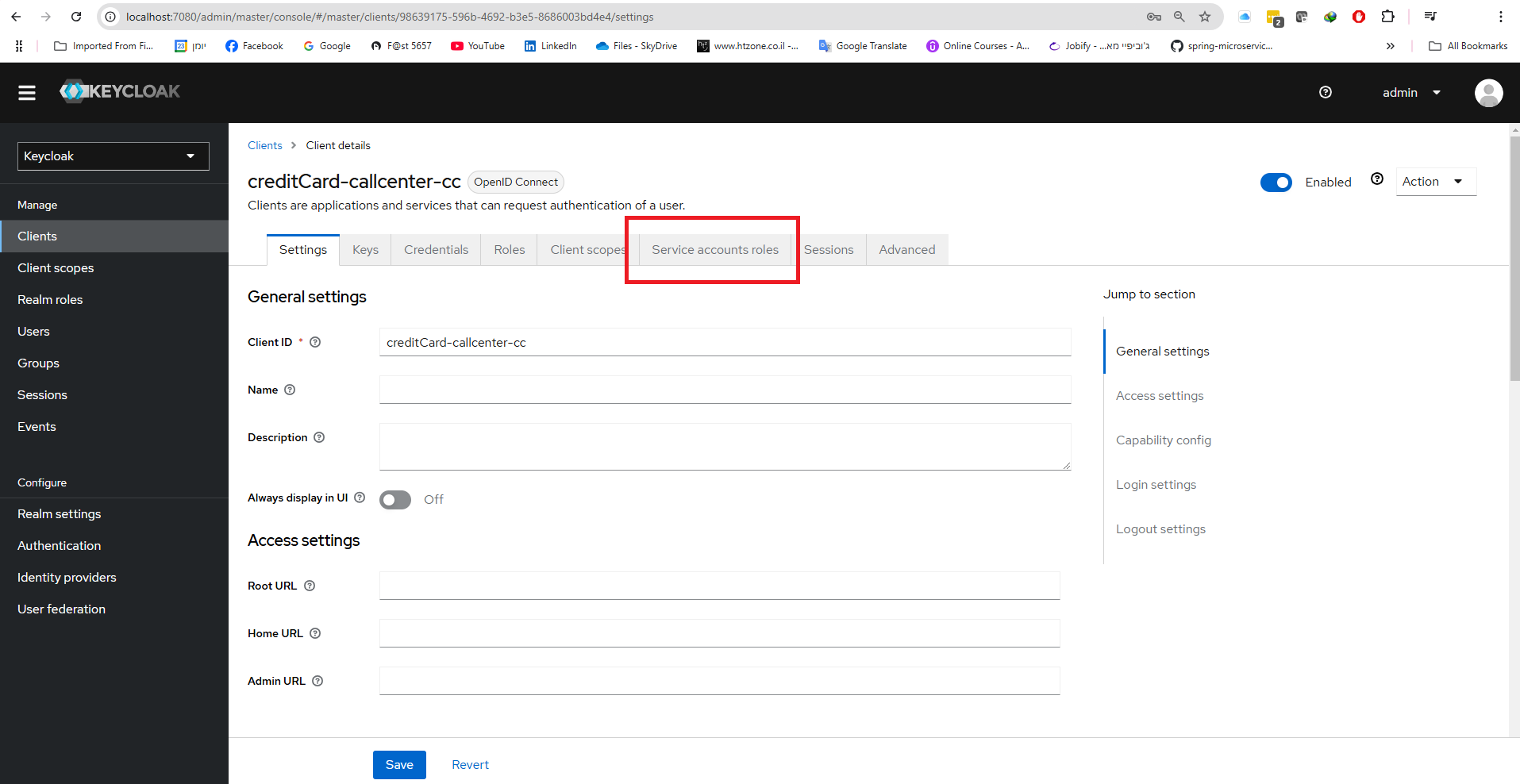
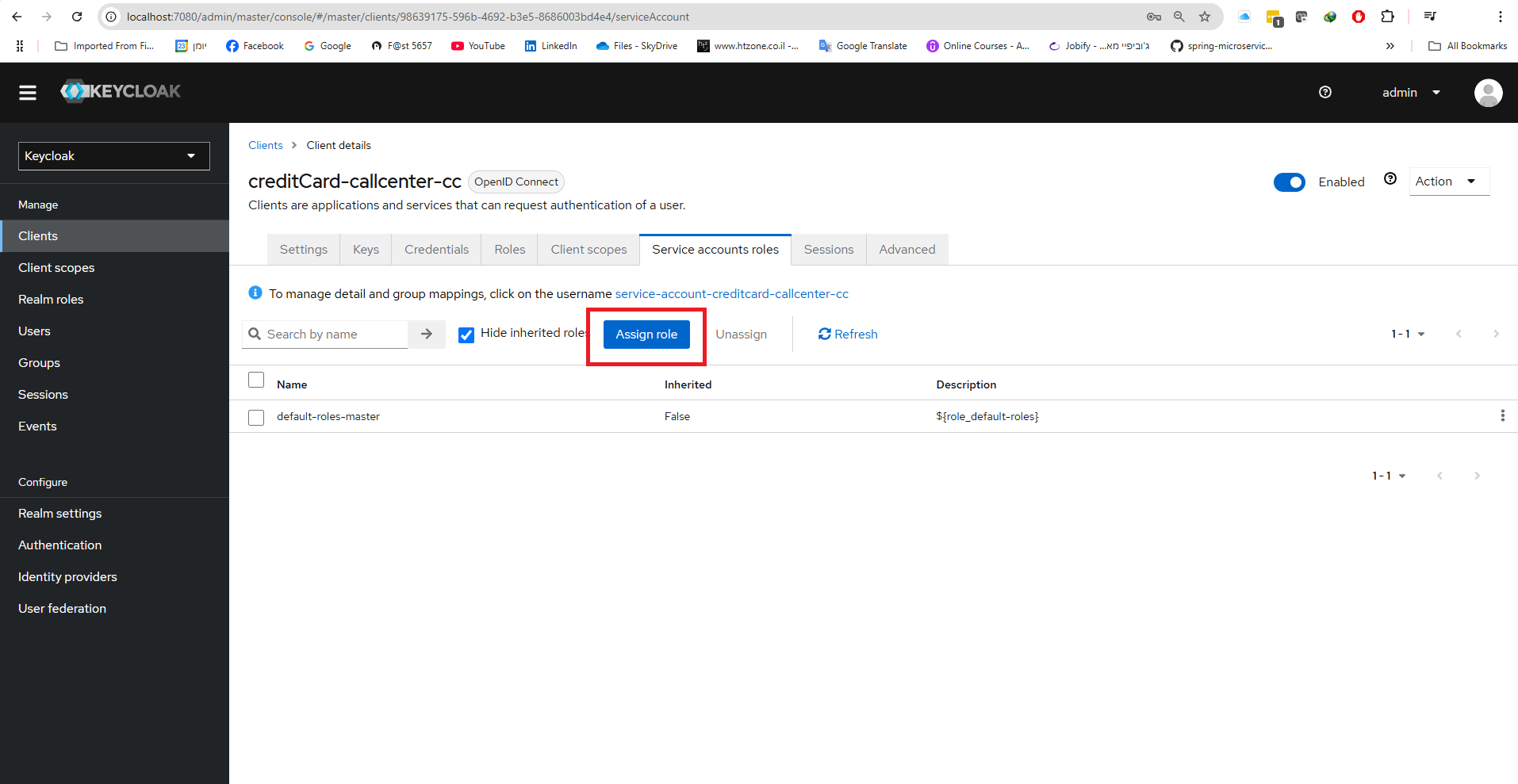


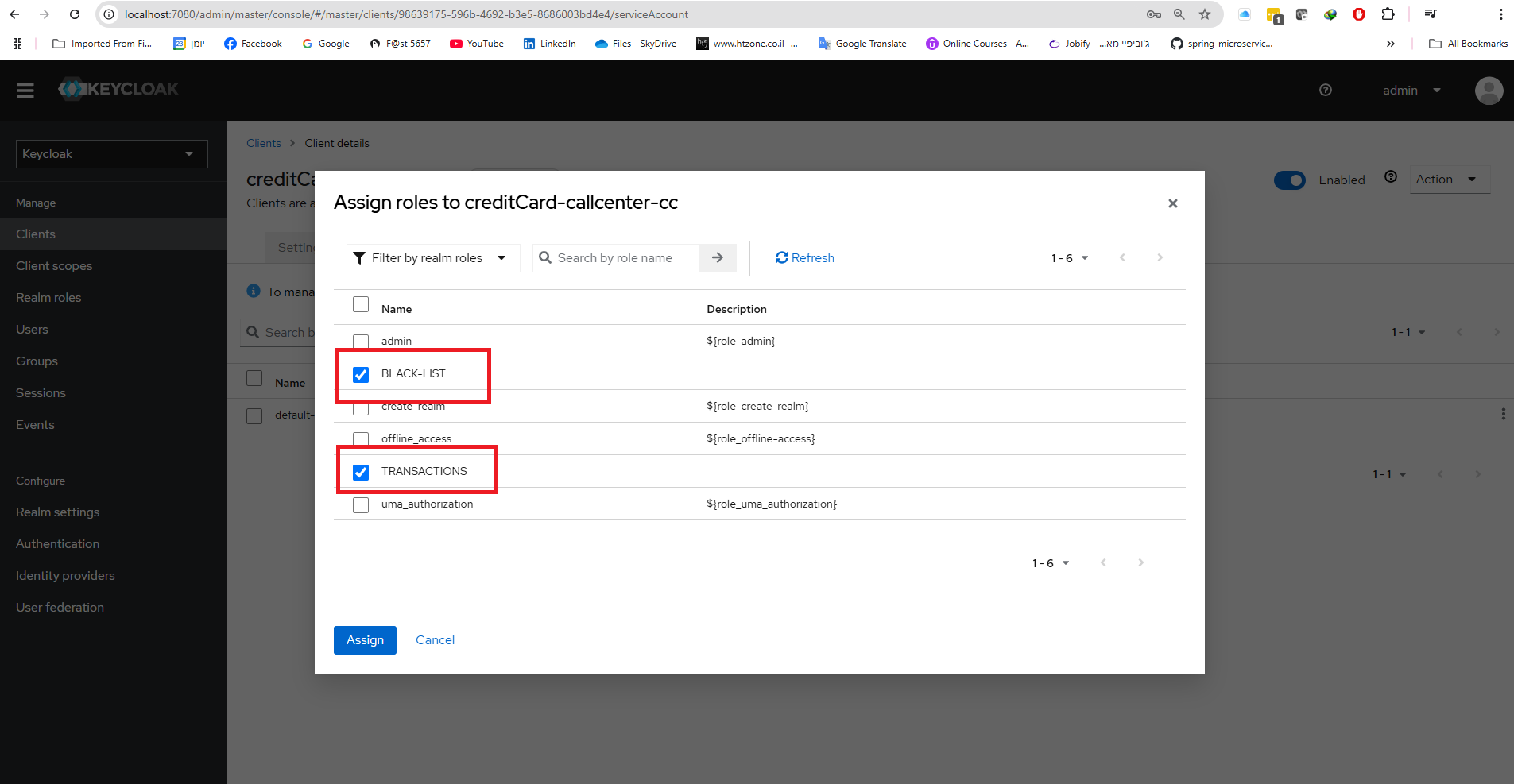




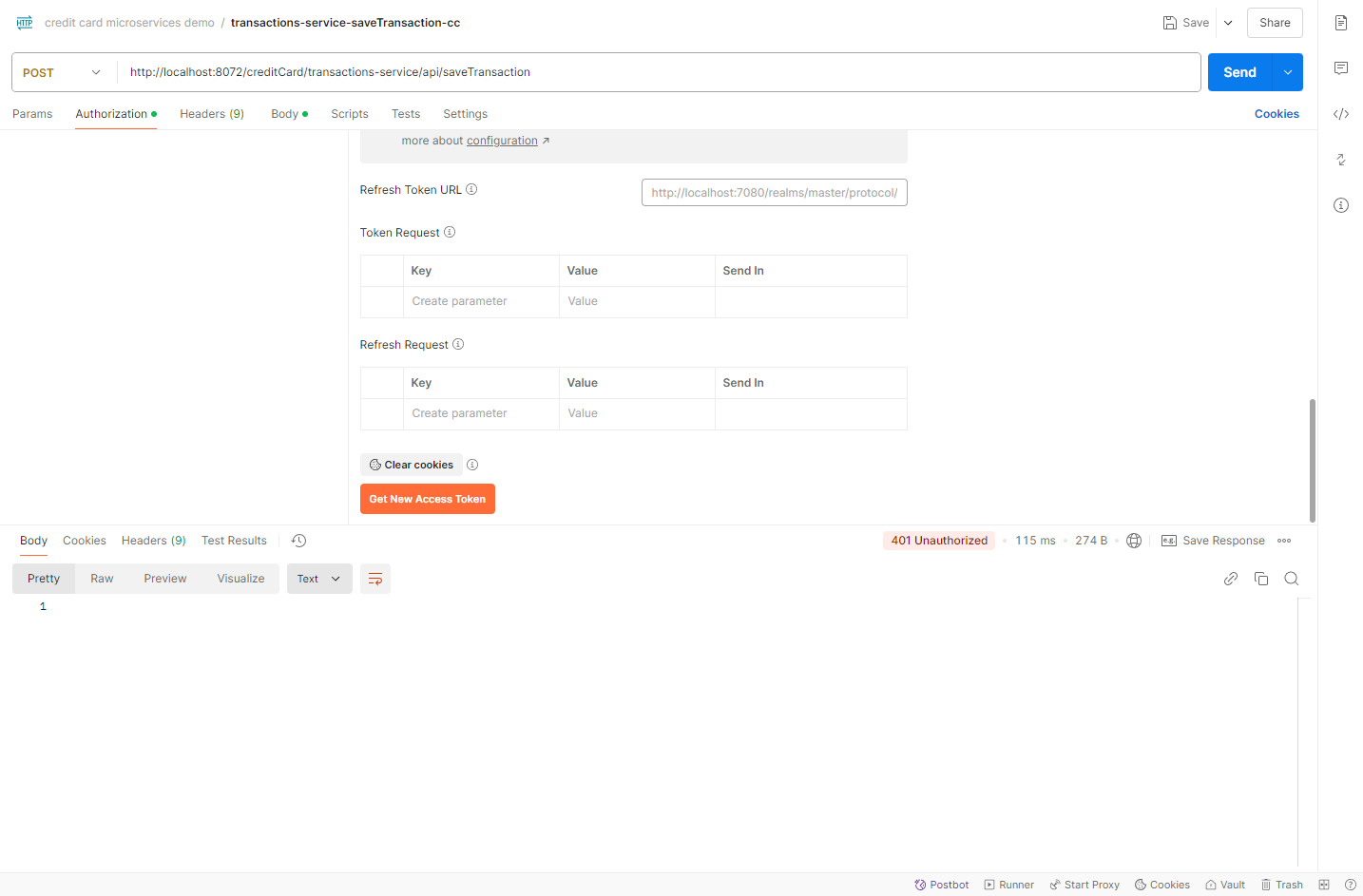
2. **Assign Roles to the Client:**

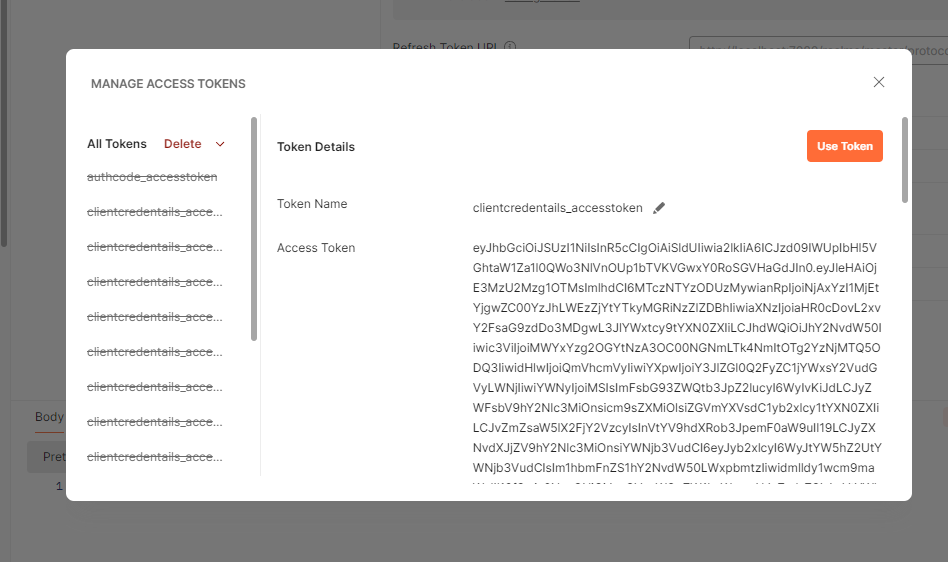
* In the left sidebar, click **Clients**, and select the creditCard-callcenter-cc client.
* Go to **Service Account Roles** and click **Assign Role**.
* Assign both the **TRANSACTIONS** and **BLACK-LIST** roles.



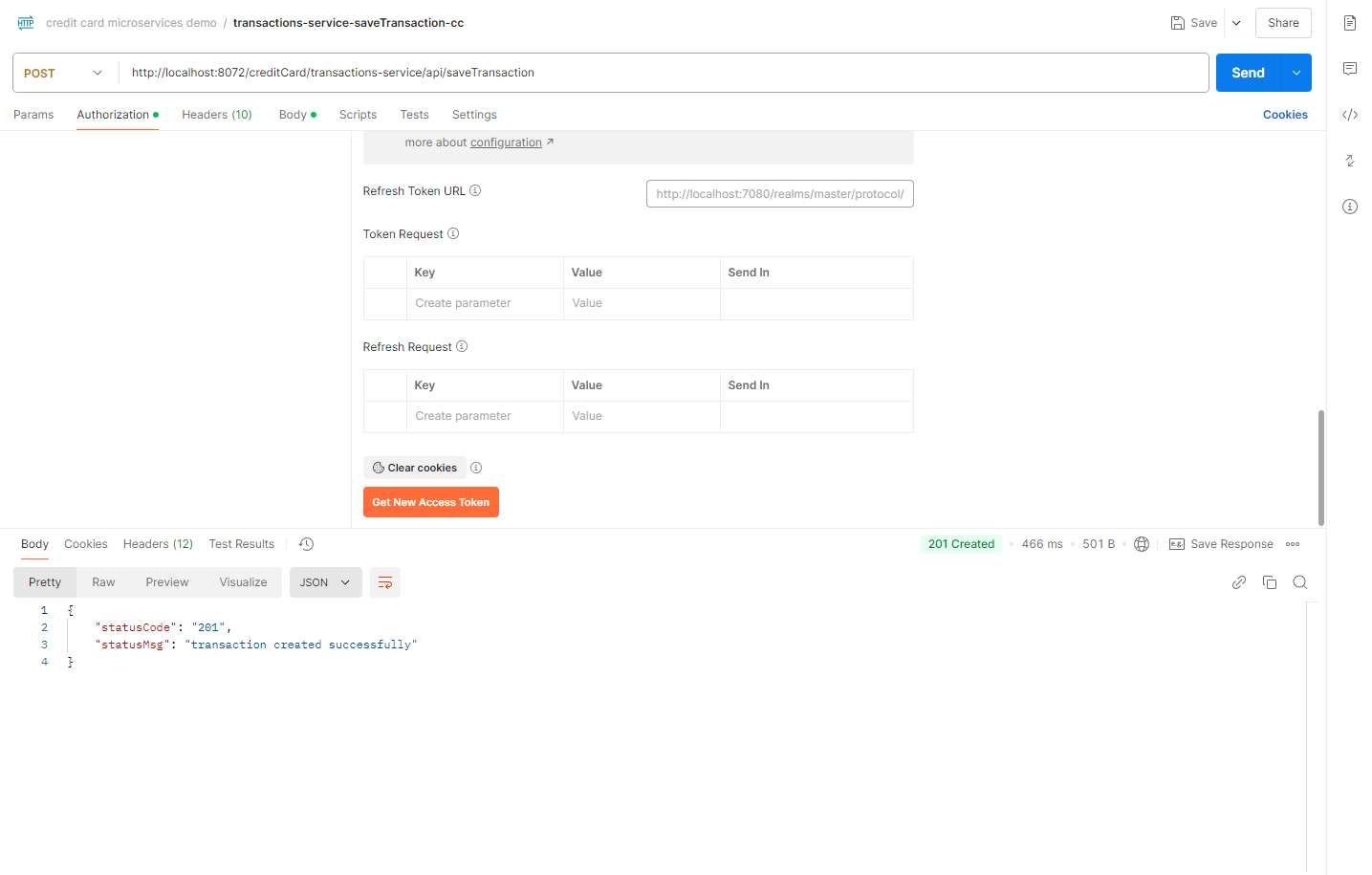


1. **Reattempt to Get Access Token:**

* In Postman, click on **Get New Access Token**.
* Click **Use Token**.

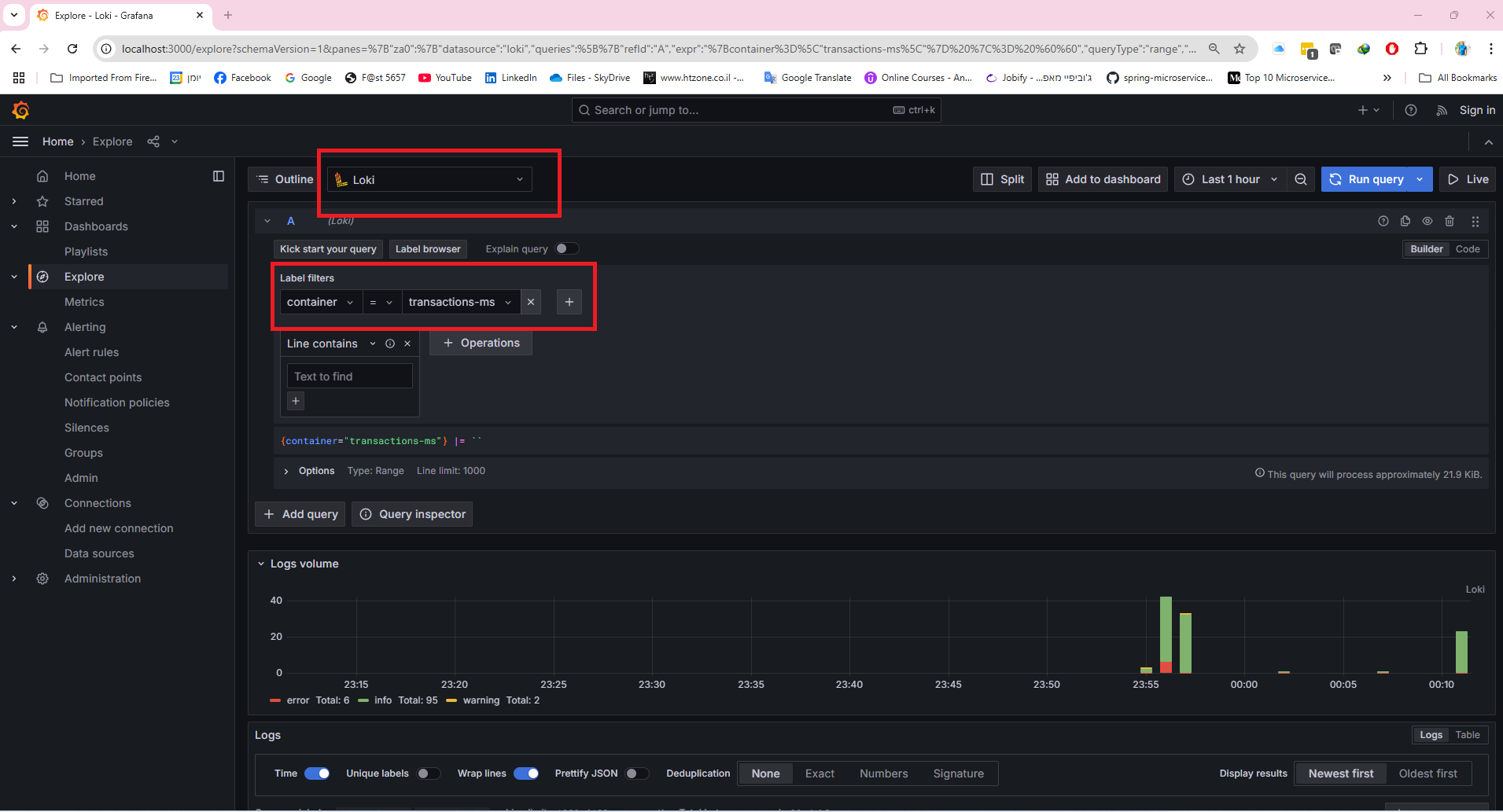


1. **Run the Request Again:**

* Now, you should receive a **201 Created** response, indicating that the transaction was successfully saved.

## Grafana

1. **Open the Application:**

* Open your browser and go to: <http://localhost:3000/>
* Once the page loads, navigate to the **Explore** section and select **Loki** to explore the logs generated by the services running in the Docker containers.

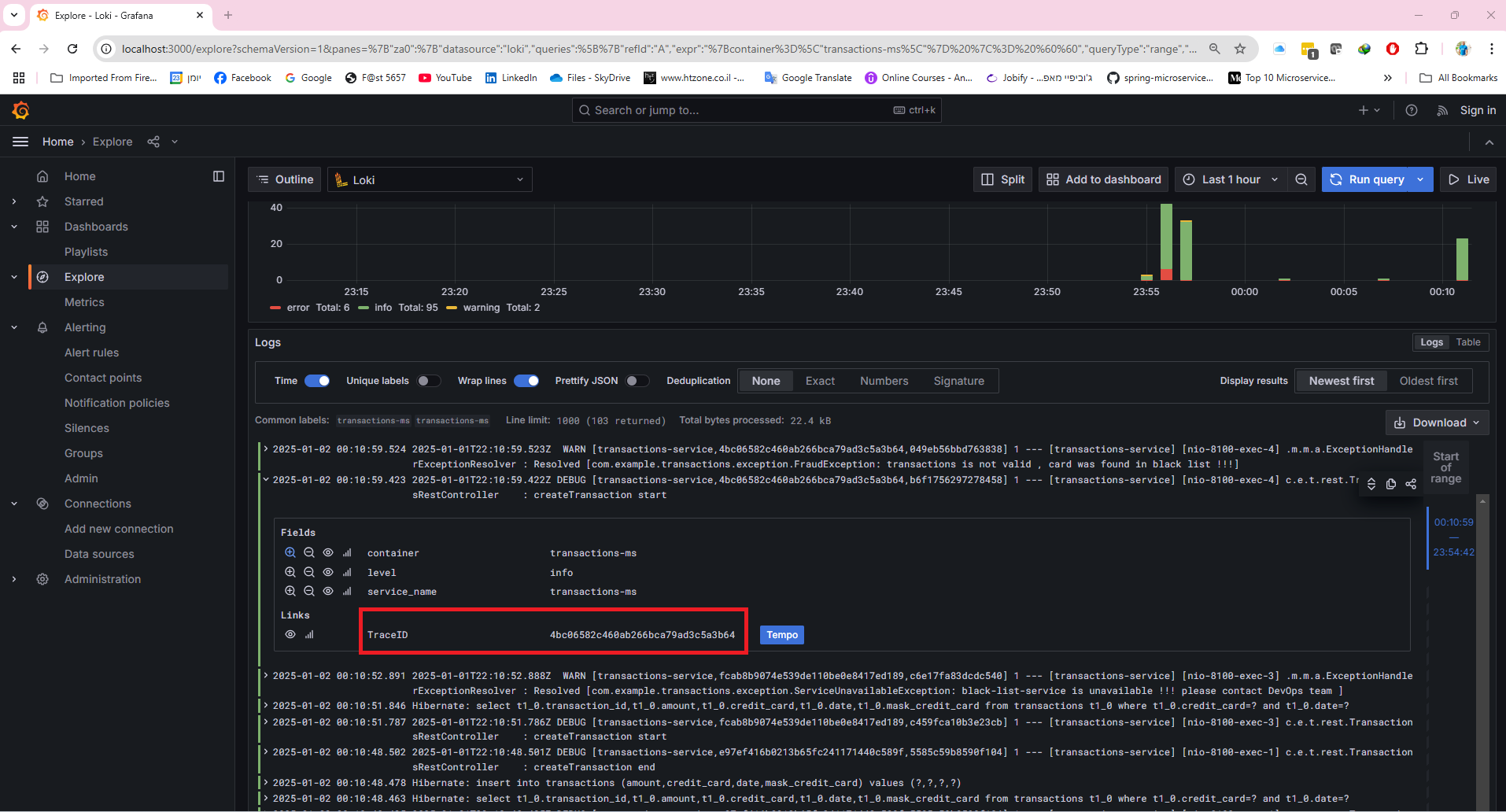
1. **Set up Tempo**
2. **Access Logs:**

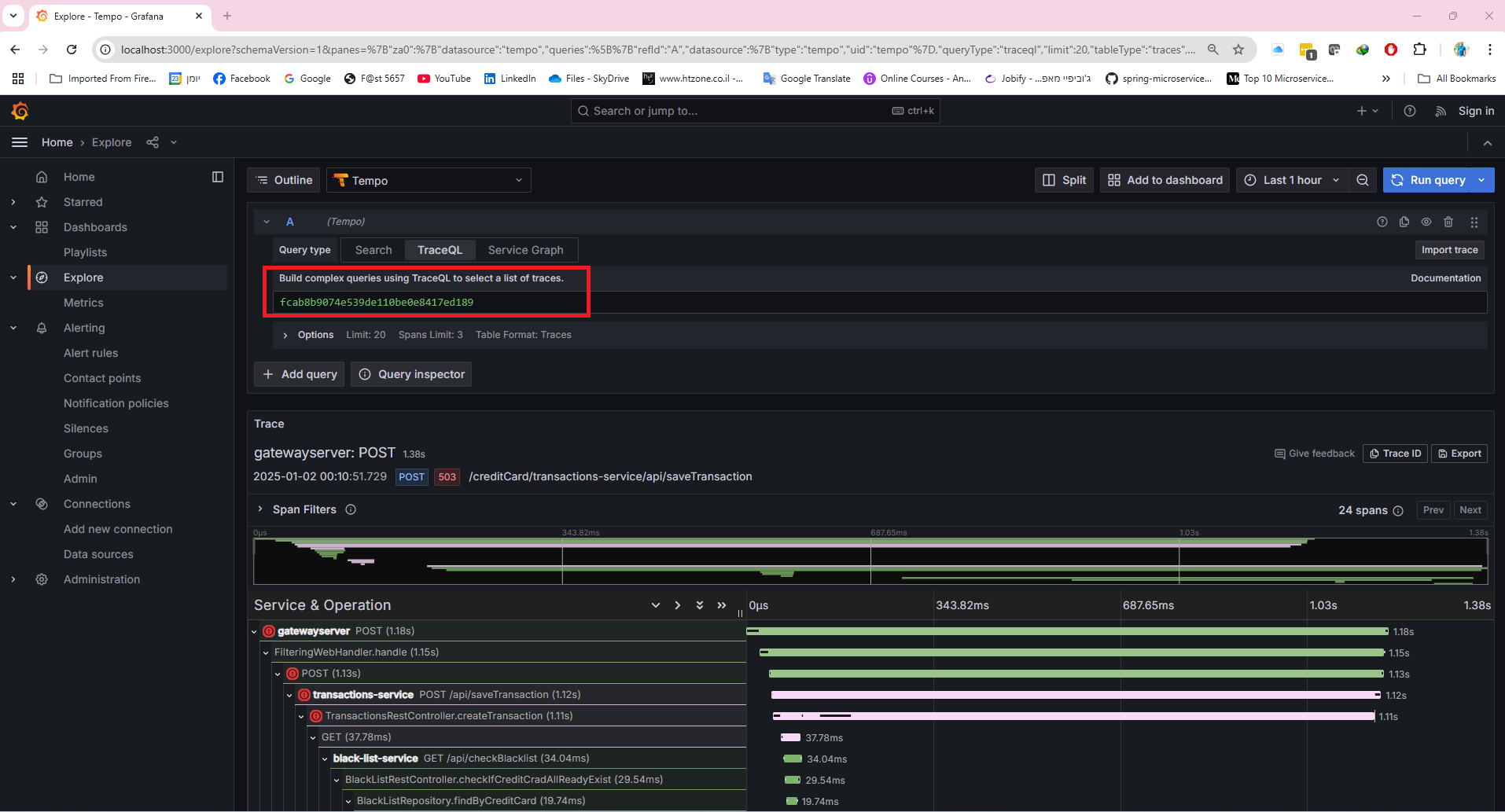
* Open one of the logs generated by the application.
* Copy the **traceID** from the log entry.

1. **Search in Tempo:**

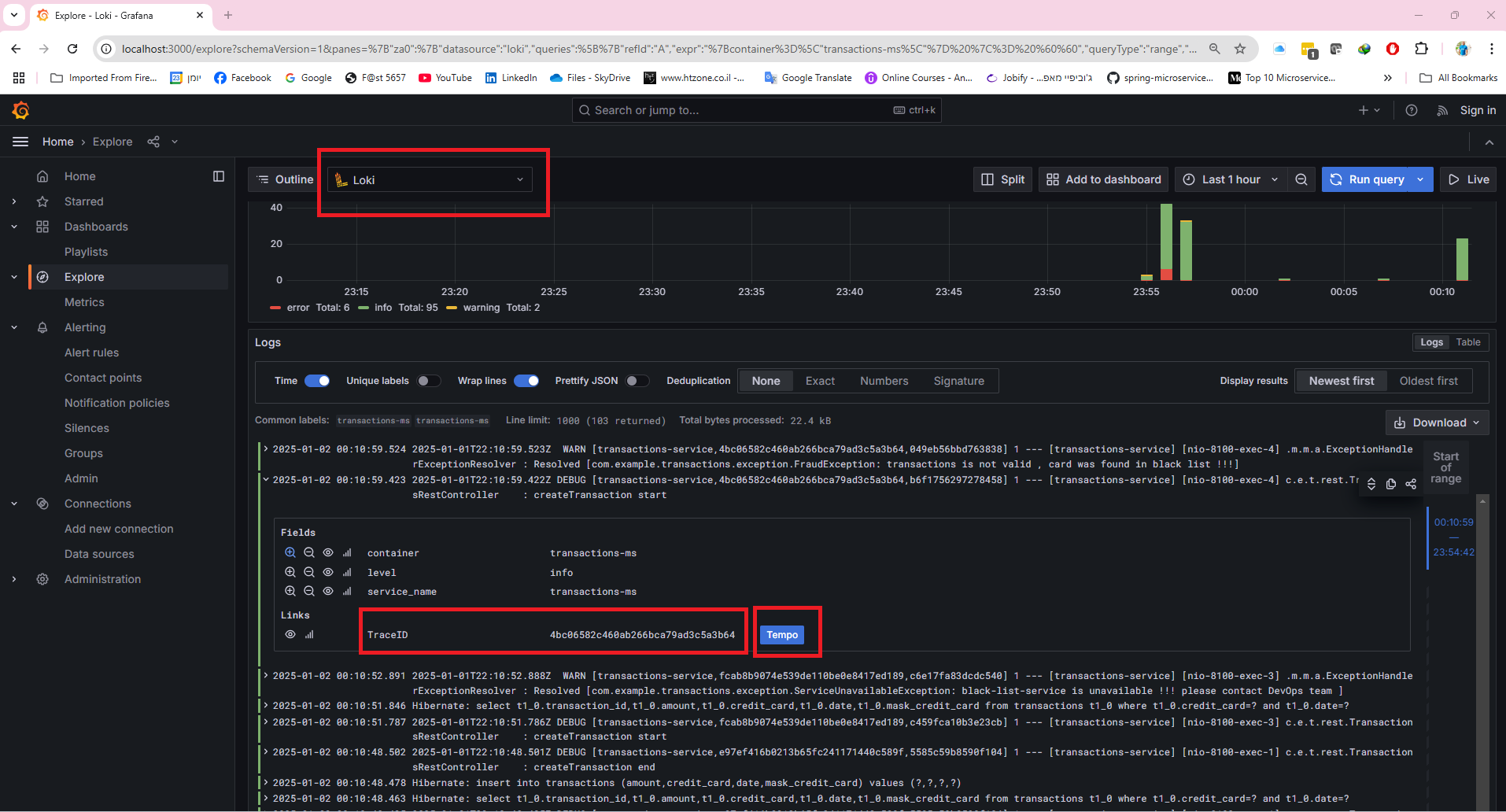
* Navigate to **Tempo**
* In Tempo's search bar, **paste the traceID** you copied. This will allow you to trace the request across various microservices involved.

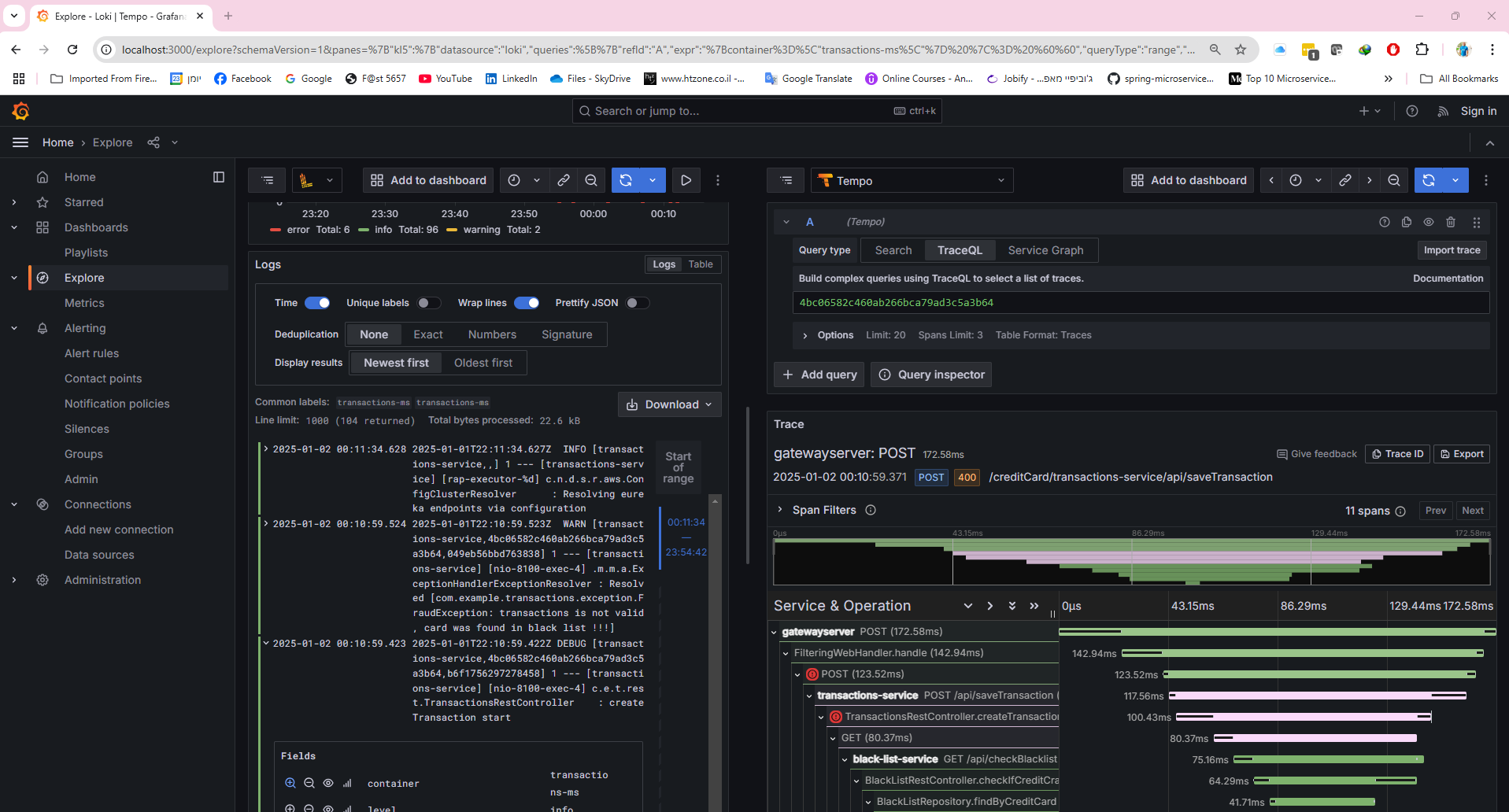
1. **View Trace Details:**

* Once the traceID is entered, Tempo will display a detailed trace showing the microservices that participated in the request, allowing you to track its flow across the system.



1. **Access Tempo from Loki:**

* Tempo can also be accessed directly from **Loki**.
* While viewing a log in Loki, look for the **Tempo button** (usually located next to the log entry).
* Click on the **Tempo button** to automatically open the trace associated with that request and view the trace details.



1. Set Up Prometheus Dashboard:

* In Grafana, go to **Dashboards** and click **New > Import**.
* Use the following link to import the **JVM Micrometer** dashboard:

<https://grafana.com/grafana/dashboards/4701-jvm-micrometer/>

* Select **Prometheus** as the data source.

