

Configuraciones iniciales de Keycloak

Rodrigo Batista - Parcial Backend II

1. Ejecutar Keycloak con Docker

```
docker run -p 8080:8080 -e KEYCLOAK_ADMIN=admin -e KEYCLOAK_ADMIN_PASSWORD=admin  
quay.io/keycloak/keycloak:22.0.1 start-dev
```

2. Crear un nuevo reino importando el archivo realm-export.json

Create realm

A realm manages a set of users, credentials, roles, and groups. A user belongs to and logs into a realm. Realms are isolated from one another and can only manage and authenticate the users that they control.

Resource file

Drag a file here or browse to upload

Browse... Clear

```
1 {  
2   "id": "5c7895b8-3a62-4d0d-a897-e1393d455775",  
3   "realm": "dh_bills",  
4   "notBefore": 0,  
5   "defaultSignatureAlgorithm": "RS256",  
6   "revokeRefreshToken": false,  
7   "refreshTokenMaxReuse": 0.  
}
```

Upload a JSON file

Realm name *

dh_bills

Enabled

☒ On

Create

Cancel

3. Ingresar al cliente "gateway-client" y regenerar el Client secret.

Clients > Client details

bills-client OpenID Connect Enabled ⓘ Action ▾

Clients are applications and services that can request authentication of a user.

Settings

Keys

Credentials

Roles

Client scopes

Service accounts roles

Sessions

Advanced

Client Authenticator ⓘ

Client Id and Secret ▾

Save

Client secret

..... ⓘ ⓘ

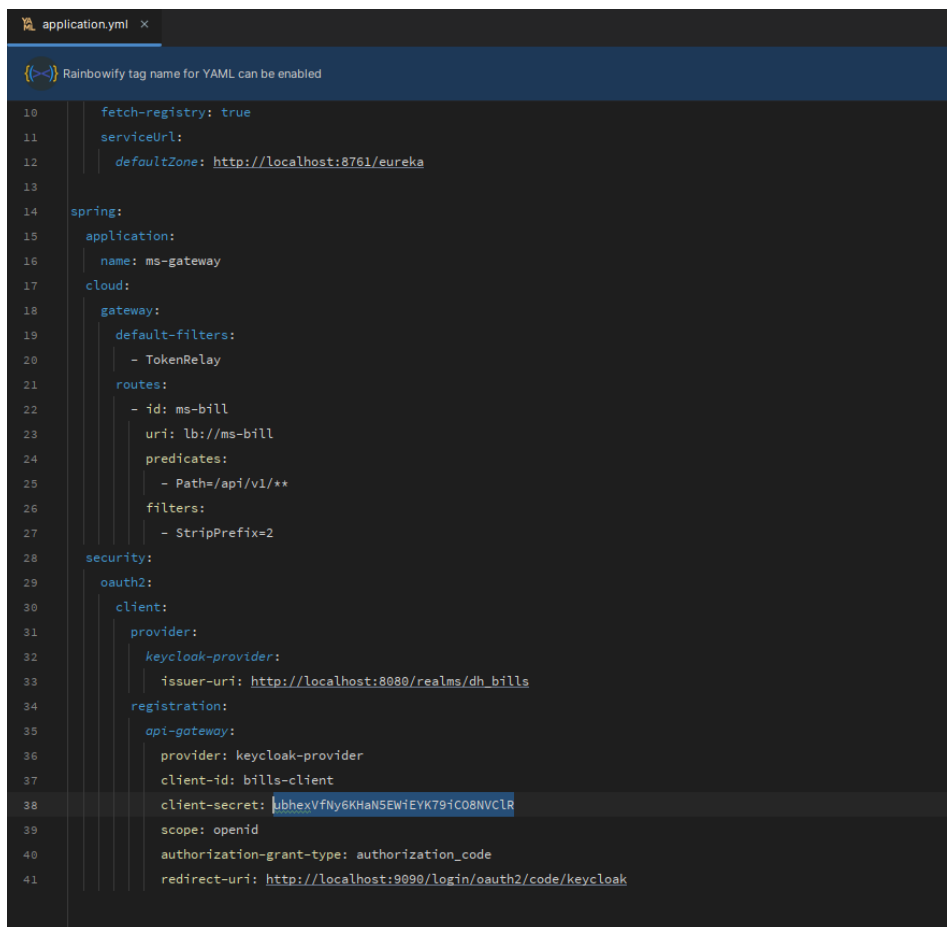
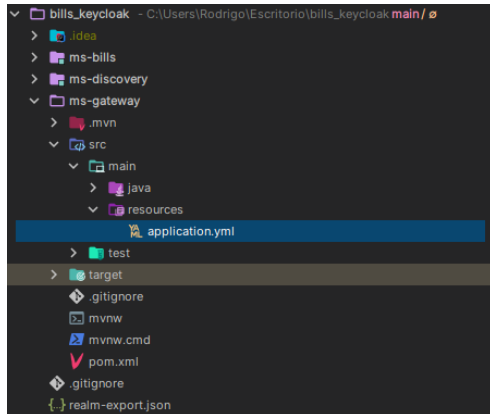
Regenerate

Registration access token ⓘ

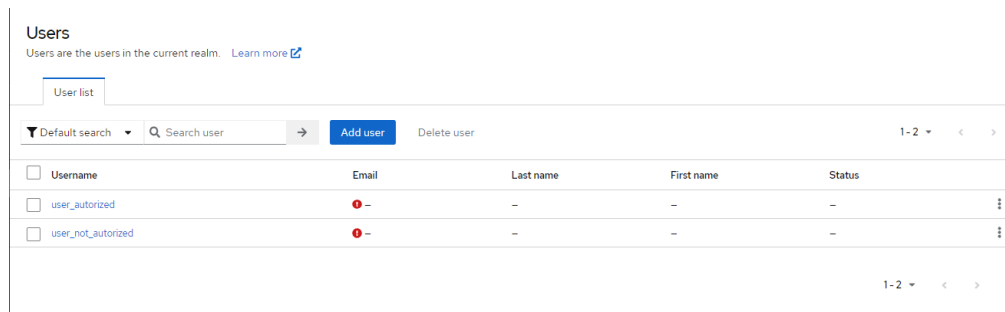
..... ⓘ

Regenerate

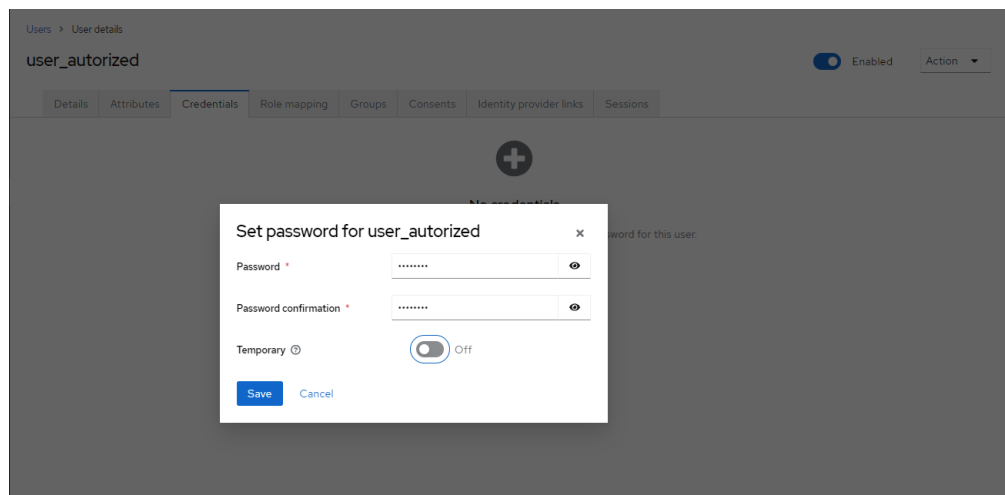
Este nuevo “client-secret” generado debe ser colocado en el application.yml correspondiente al microservicio del gateway.



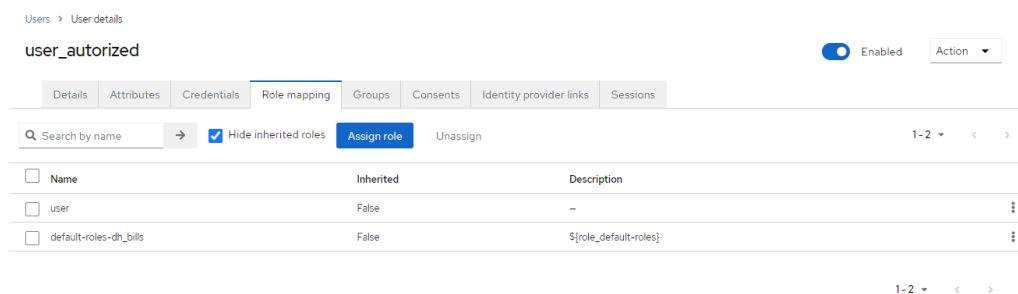
- Hacer lo mismo con el cliente “user-client”, y colocar el nuevo “client-secret” generado en el application.yml correspondiente al microservicio de users.
- Crear tres usuarios en el reino: user_authorized, user_not_authorized, user_provider



- Asignar una contraseña a cada uno de los usuarios creados (desmarcar que sean temporales)



- Asignar el rol de “user” (rol a nivel de reino), solo al usuario “user_authorized”



8. Asignar el grupo de “providers” solo al usuario de “user_provider”

Users > User details

user_provider

DetailsAttributesCredentialsRole mappingGroupsConsentsIdentity provider linksSessions

Search group

→

Join Group

☒ Direct membership

Leave

☒ Who will appear in this group list?

<input type="checkbox"/> Group membership	Path
<input type="checkbox"/> providers	/providers

9. El usuario “user_not_authorized” debe quedar sin el rol user asignado.

Users > User details

user_not_authorized

DetailsAttributesCredentialsRole mappingGroupsConsentsIdentity provider linksSessions

Search by name

→

☒ Hide inherited roles

Assign role

Unassign

1-1

<

>

<input type="checkbox"/> Name	Inherited	Description
<input type="checkbox"/> default-roles-dh_bills	False	\${role_default-roles}

1-1<>

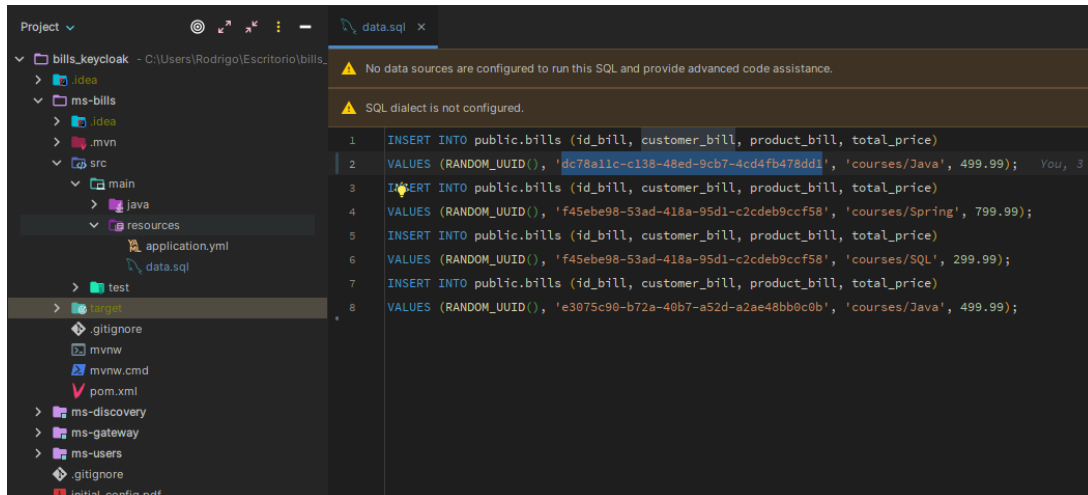
10. Crear 2 usuarios más con los siguientes datos:

```
{
  username: johndoe
  email: johndoe@mail.com
  emailVerified: true
  firstName: John
  lastName: Doe
},
{
  username: janedoe
  email: janedoe@mail.com
  emailVerified: true
  firstName: Jane
  lastName: Doe
}
```

11. Ingresar a un usuario de los creados (ej. "johndoe") y copiar el id que genera keycloak.

Ingresar al archivo ("data.sql") que carga los datos iniciales en el microservicio de bills y pegar este id en la columna de "customer_bills" de los 3 primeros registros.

En la misma columna pero del último registro, pegar el id del segundo usuario creado ("janedoe").



```
1 INSERT INTO public.bills (id_bill, customer_bill, product_bill, total_price)
2 VALUES (RANDOM_UUID(), 'dc78a11c-c138-48ed-9cb7-4cd4fb478dd1', 'courses/Java', 499.99);
3 INSERT INTO public.bills (id_bill, customer_bill, product_bill, total_price)
4 VALUES (RANDOM_UUID(), 'f45ebe98-53ad-418a-95d1-c2cdeb9ccf58', 'courses/Spring', 799.99);
5 INSERT INTO public.bills (id_bill, customer_bill, product_bill, total_price)
6 VALUES (RANDOM_UUID(), 'f45ebe98-53ad-418a-95d1-c2cdeb9ccf58', 'courses/SQL', 299.99);
7 INSERT INTO public.bills (id_bill, customer_bill, product_bill, total_price)
8 VALUES (RANDOM_UUID(), 'e3075c90-b72a-40b7-a52d-a2ae48bb0c0b', 'courses/Java', 499.99);
```