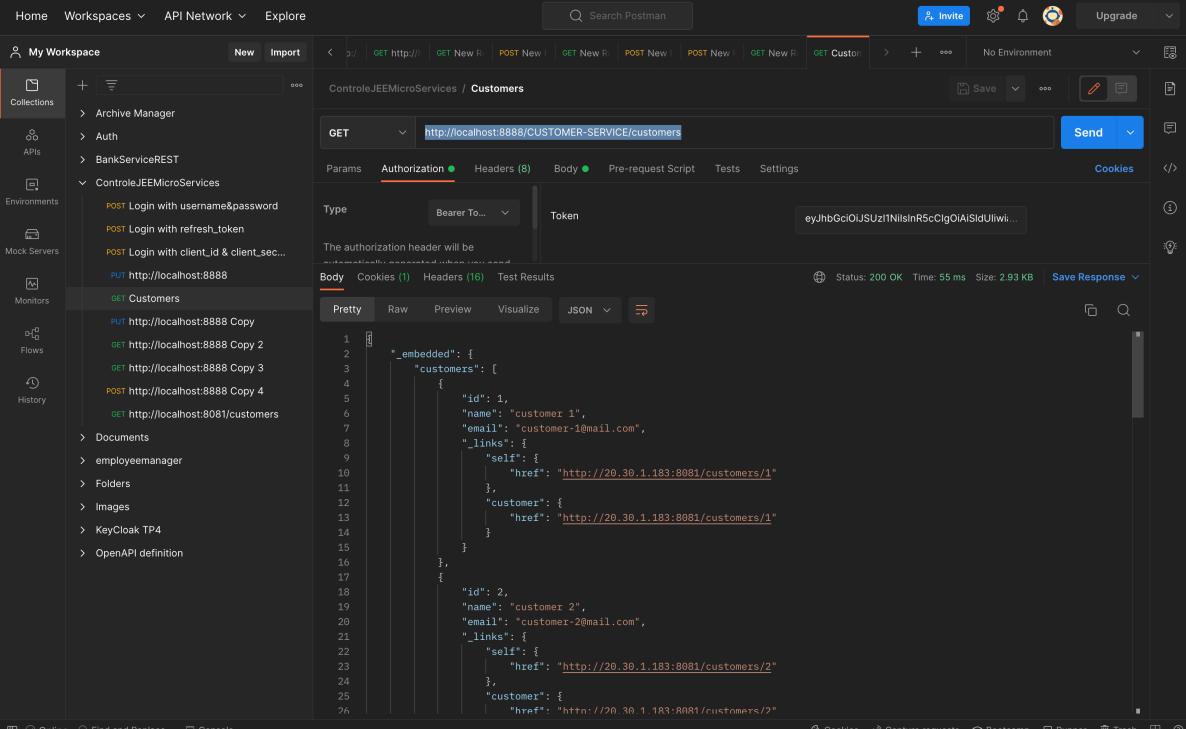


Controle JEE Rapport

1. Customer Service

Get all customers



The screenshot shows the Postman application interface. On the left, there's a sidebar with various sections like Collections, APIs, Environments, Mock Servers, Monitors, Flows, and History. The main area shows a collection named "ControleJEEMicroServices" with a "Customers" endpoint. A GET request is made to `http://localhost:8888/CUSTOMER-SERVICE/customers`. The "Authorization" tab is selected, showing a token header with a placeholder value. The "Body" tab displays the JSON response:

```
1  {
2   "embedded": {
3     "customers": [
4       {
5         "id": 1,
6         "name": "customer 1",
7         "email": "customer1@mail.com",
8         "links": {
9           "self": {
10             "href": "http://20.30.1.183:8081/customers/1"
11           },
12           "customer": {
13             "href": "http://20.30.1.183:8081/customers/1"
14           }
15         }
16       },
17       {
18         "id": 2,
19         "name": "customer 2",
20         "email": "customer2@mail.com",
21         "links": {
22           "self": {
23             "href": "http://20.30.1.183:8081/customers/2"
24           },
25           "customer": {
26             "href": "http://20.30.1.183:8081/customers/2"
27           }
28         }
29       }
30     ]
31   }
32 }
```

Add new costumer

The screenshot shows the Postman interface with the following details:

- Collection:** My Workspace
- Request:**
 - Method: POST
 - URL: http://localhost:8888/CUSTOMER-SERVICE/customers
 - Body (JSON):

```

1
2   ...
3     "name": "abdelalim edited",
4     "email": "abdelalim@mail.com"
      
```
- Response:**
 - Status: 201 Created
 - Time: 19 ms
 - Size: 788 B

Edit customer 10

The screenshot shows the Postman interface with the following details:

- Collection:** My Workspace
- Request:**
 - Method: PUT
 - URL: http://localhost:8888/CUSTOMER-SERVICE/customers/10
 - Body (JSON):

```

1
2   ...
3     "name": "abdelalim edited one more time",
4     "email": "abdelalim-edited@mail.com"
      
```
- Response:**
 - Status: 200 OK
 - Time: 22 ms
 - Size: 804 B

Delete costumer 10

The screenshot shows the Postman interface with the following details:

- Left Sidebar:** My Workspace, Collections, APIs, Environments, Mock Servers, Monitors, Flows, History.
- Request Details:** ControleJEEMicroServices / Customers, DELETE, http://localhost:8888/CUSTOMER-SERVICE/customers/10.
- Body:** JSON, containing:

```
1
2 ... "name": "abdelalim edited one more time",
3 "email": "abdelalim-edited@mail.com"
4
```
- Response:** Status: 204 No Content, Time: 13 ms, Size: 424 B.

Check if customer 10 exists

The screenshot shows the Postman interface with the following details:

- Left Sidebar:** My Workspace, Collections, APIs, Environments, Mock Servers, Monitors, Flows, History.
- Request Details:** ControleJEEMicroServices / Customers, GET, http://localhost:8888/CUSTOMER-SERVICE/customers/10.
- Body:** JSON, containing:

```
1
2 ... "name": "abdelalim edited one more time",
3 "email": "abdelalim-edited@mail.com"
4
```
- Response:** Status: 404 Not Found, Time: 18 ms, Size: 442 B.

2. Inventory Service

Gell all products

The screenshot shows the Postman interface with the following details:

- Workspace:** My Workspace
- Request:** GET http://localhost:8888/INVENTORY-SERVICE/products
- Headers:** Authorization (set to Bearer [redacted])
- Body:** PRETTY JSON response (Status: 200 OK, Time: 37 ms, Size: 1.82 KB)
- Response:** A JSON object representing a collection of products. The first product is shown in detail:

```
1  "embedded": {  
2      "products": [  
3          {  
4              "id": 1,  
5              "name": "Product 1",  
6              "price": 1500.0,  
7              "quantity": 200,  
8              "_links": {  
9                  "self": {  
10                     "href": "http://20.30.1.183:8082/products/1"  
11                 },  
12                  "product": {  
13                     "href": "http://20.30.1.183:8082/products/1"  
14                 }  
15             },  
16         },  
17         {  
18             "id": 2,  
19             "name": "Product 2"  
20         }  
21     ]  
22 }
```

Add new product

POST POST Test

ControleJEEMicroServices / POST Test

POST http://localhost:8888/INVENTORY-SERVICE/products

```

1
2   "name": "Test product",
3   "price": 2000,
4   "quantity": 50
5
6
7
8
9
10
11
12
13
14

```

Status: 201 Created Time: 21 ms Size: 781 B Save Response

Edit product 5

PUT POST Test

ControleJEEMicroServices / POST Test

PUT http://localhost:8888/INVENTORY-SERVICE/products/5

```

1
2   "name": "Test product edited",
3   "price": 2000,
4   "quantity": 50
5
6
7
8
9
10
11
12
13
14

```

Status: 200 OK Time: 19 ms Size: 783 B Save Response

Delete product 5

The screenshot shows the Postman application interface. On the left, there's a sidebar with sections like 'Collections', 'APIs', 'Environments', 'Mock Servers', 'Monitors', 'Flows', and 'History'. Under 'Collections', there's a tree view with 'ControleJEEMicroServices' expanded, showing various API endpoints such as 'POST Login with username&password', 'POST Login with refresh_token', 'POST Login with client_id & client_sec...', 'PUT POST Test', 'GET Customers', 'GET Bills', 'GET Products', and several 'Copy' requests. The main workspace shows a 'DELETE' request to 'http://localhost:8888/INVENTORY-SERVICE/products/5'. The 'Body' tab is selected, displaying a JSON payload:

```
1
2   "name": "Test product edited",
3   "price": 2000,
4   "quantity": 50
```

Below the request, the status bar indicates 'Status: 204 No Content Time: 24 ms Size: 424 B'. At the bottom, there are tabs for 'Pretty', 'Raw', 'Preview', 'Visualize', 'Text', and a 'Send' button.

Check if product 5 exists

This screenshot is similar to the previous one but shows a 'GET' request instead. The URL is 'http://localhost:8888/INVENTORY-SERVICE/products/5'. The response status is 'Status: 404 Not Found Time: 17 ms Size: 442 B'. The rest of the interface and payload are identical to the previous screenshot.

3. Gateway Service with dynamic routes

BILLING-SERVICE : localhost:8888/BILLING-SERVICE/bills

The screenshot shows the Postman application interface. On the left, the 'My Workspace' sidebar is open, displaying collections, environments, mock servers, monitors, flows, and history. The 'Collections' section is expanded, showing a tree structure with 'Archive Manager', 'Auth', 'BankServiceREST', and 'ControladorJEEMicroServices'. Under 'ControladorJEEMicroServices', several API endpoints are listed: 'Login with username&password' (POST), 'Login with refresh_token' (POST), 'Login with client_id & client_sec...' (POST), 'http://localhost:8888' (PUT), 'Customers' (GET), 'Bills' (GET), 'Products' (GET), 'http://localhost:8888 Copy' (PUT), 'http://localhost:8888 Copy 2' (GET), 'http://localhost:8888 Copy 3' (GET), 'http://localhost:8888 Copy 4' (POST), and 'http://localhost:8081/customers' (GET). The main workspace shows a request for 'Bills' with the URL `http://localhost:8888/BILLING-SERVICE/bills`. The 'Params' tab is selected, showing a single parameter 'Key' with value 'Value'. The 'Body' tab is selected, showing a JSON response structure. The response body is as follows:

```
1  {
2   "embedded": {
3     "bills": []
4   },
5   "_links": {
6     "self": {
7       "href": "http://20.30.1.183:8083/bills"
8     },
9     "profile": {
10       "href": "http://20.30.1.183:8083/profile/bills"
11     },
12     "search": {
13       "href": "http://20.30.1.183:8083/bills/search"
14     }
15   },
16   "page": {
17     "size": 20,
18     "totalElements": 0,
19     "totalPages": 0,
20     "numberOfPages": 0
21   }
22 }
```

The status bar at the bottom indicates: Status: 200 OK, Time: 42 ms, Size: 943 B, Save Response, and a search bar for 'Find and Replace'.

BILLING-SERVICE : localhost:8888/CUSTOMER-SERVICE/customers

ControleJEEMicroServices / Customers

GET http://localhost:8888/CUSTOMER-SERVICE/customers

```

1
2   "_embedded": {
3     "customers": [
4       {
5         "id": 1,
6         "name": "customer 1",
7         "email": "customerx-1@mail.com",
8         "_links": {
9           "self": {
10             "href": "http://20.30.1.183:8081/customers/1"
11           },
12           "customer": {
13             "href": "http://20.30.1.183:8081/customers/1"
14           }
15         }
16       },
17       {
18         "id": 2,
19         "name": "customer 2",
20         "email": "customerx-2@mail.com",
21         "_links": {
22           "self": {
23             "href": "http://20.30.1.183:8081/customers/2"
24           },
25           "customer": {
26             "href": "http://20.30.1.183:8081/customers/2"
27           }
28         }
29       }
30     ]
31   }
32 }
```

BILLING-SERVICE : localhost:8888/INVENTORY-SERVICE/products

ControleJEEMicroServices / Customers

GET http://localhost:8888/INVENTORY-SERVICE/products

```

1
2   "_embedded": {
3     "products": [
4       {
5         "id": 1,
6         "name": "Product 1",
7         "price": 1500.0,
8         "quantity": 200,
9         "_links": {
10           "self": {
11             "href": "http://20.30.1.183:8082/products/1"
12           },
13           "product": {
14             "href": "http://20.30.1.183:8082/products/1"
15           }
16         }
17       },
18       {
19         "id": 2,
20         "name": "Product 2",
21         "price": 3000.0,
22         "quantity": 500,
23         "_links": {
24           "self": {
25             "href": "http://20.30.1.183:8082/products/2"
26           }
27         }
28       }
29     ]
30   }
31 }
```

4. Discovery Service

The screenshot shows the Spring Eureka dashboard running on localhost:8761. The top navigation bar includes links for 'Front' (disabled), 'Activité Pratique du Contrôle', 'Eureka', 'HOME', and 'LAST 1000 SINCE STARTUP'. The main content area is divided into sections: 'System Status' and 'DS Replicas'.

System Status

Environment	test	Current time	2022-12-22T09:23:54 +0100
Data center	default	Uptime	00:20
		Lease expiration enabled	true
		Renews threshold	10
		Renews (last min)	20

DS Replicas

localhost

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
BILLING-SERVICE	n/a (1)	(1)	UP (1) - 192.168.100.28:billing-service:8083
CONFIG-SERVER	n/a (1)	(1)	UP (1) - 192.168.100.28:config-server:8088
CUSTOMER-SERVICE	n/a (1)	(1)	UP (1) - 192.168.100.28:customer-service:8081
GATEWAY-SERVICE	n/a (1)	(1)	UP (1) - 192.168.100.28:gateway-service:8888
INVENTORY-SERVICE	n/a (1)	(1)	UP (1) - 192.168.100.28:inventory-service:8082

General Info

5. Billing Service

Get all bills

My Workspace

ControleJEEMicroServices / Bills

GET http://localhost:8888/BILLING-SERVICE/bills

Key	Value	Description	Bulk Edit

```

1 {
2   "_embedded": {
3     "bills": []
4   },
5   "_links": [
6     "self": {
7       "href": "http://20.30.1.183:8083/bills"
8     },
9     "profile": {
10       "href": "http://20.30.1.183:8083/profile/bills"
11     },
12     "search": {
13       "href": "http://20.30.1.183:8083/bills/search"
14     }
15   },
16   "page": {
17     "size": 20,
18     "totalElements": 0,
19     "totalPages": 0,
20     "number": 0
21   }
22 }

```

Add new bill

My Workspace

ControleJEEMicroServices / Bills

POST http://localhost:8888/BILLING-SERVICE/fullBill

Body

```

1 {
2   "billingDate": "2022-12-24T00:00:00.000+00:00",
3   "productItems": [
4     {
5       "quantity": 6,
6       "productID": 2,
7       "price": 3000,
8       "product": {
9         "id": 2,
10        "name": "Product 2",
11        "price": 3000,
12        "quantity": 494
13      }
14    ],
15   "customerID": 7
16 }

```

View bill

The screenshot shows the Postman interface with the following details:

- Collection:** ControleJEEMicroServices / Bills
- Method:** GET
- URL:** http://localhost:8888/BILLING-SERVICE/fullBill/2
- Body (JSON):**

```

1
2 "id": 2,
3 "billingDate": "2022-12-24T00:00:00.000+00:00",
4 "productItems": [
5   {
6     "id": 5,
7     "quantity": 6,
8     "productID": 2,
9     "price": 3000.0,
10    "product": {
11      "id": 2,
12      "name": "Product 2",
13      "price": 3000.0,
14      "quantity": 488
15    }
16  ],
17  "customerID": 7,
18  "customer": {
19    "id": 7,
20    "name": "wahbi",
21    "email": "wahbi@mail.com"
22  }
23
24

```
- Status:** 200 OK
- Time:** 42 ms
- Size:** 652 B

Delete bill 2

The screenshot shows the Postman interface with the following details:

- Collection:** ControleJEEMicroServices / Bills
- Method:** DELETE
- URL:** http://localhost:8888/BILLING-SERVICE/fullBill/2
- Body (Text):**

```
1
```
- Status:** 200 OK
- Time:** 30 ms
- Size:** 346 B

6. Angular Client

Customers Management

The screenshot shows a web browser window for the URL `localhost:4200/user/customers`. The title bar includes the text "Activité Pratique du Contrôle" and "Eureka". The top navigation bar has links for "JEE-Contrôle", "Products", "Customers", "Bills", and "Back". On the right, it shows "admin admin". A "New customer" button is visible at the top left of the main content area. The main content is titled "Customers List" and contains a table with columns: "ID", "Customer name", "Customer email", and "Bills". The table has five rows with data: 1. customer 1, customer-1@mail.com; 2. customer 2, customer-2@mail.com; 3. customer 3, customer-3@mail.com; 4. achraf, achraf@mail.com; 5. salim, salim@mail.com. Each row in the "Bills" column contains three icons: a dollar sign (\$), a yellow pencil, and a red trash can. Below the table, there is a pagination section showing "1 / 2" and "5 per page", with buttons for "1" and "2".

ID	Customer name	Customer email	Bills
1	customer 1	customer-1@mail.com	\$ ⚡ 🗑
2	customer 2	customer-2@mail.com	\$ ⚡ 🗑
3	customer 3	customer-3@mail.com	\$ ⚡ 🗑
4	achraf	achraf@mail.com	\$ ⚡ 🗑
5	salim	salim@mail.com	\$ ⚡ 🗑

A Front		localhost:4200/user/customers	Activité Pratique du Contrôle	Eureka		
JEE-Contrôle		Products Customers Bills Back	admin admin			
New customer						
Customers List						
<input type="text" value="Keyword"/> <input type="button" value="Search"/>						
ID	Customer name	Customer email	Bills			
6	khalid	khalid@mail.com	<input type="button" value="\$"/>	<input type="button" value="✎"/>		
2 / 2 5 per page						
<input type="button" value="1"/> <input type="button" value="2"/>						

A Front		localhost:4200/user/new-customer	Activité Pratique du Contrôle	Eureka
JEE-Contrôle		Products Customers Bills Back	admin admin	
<p>Customer name <input type="text" value="wahbi"/></p> <p>Customer email <input type="text" value="wahbi@mail.com"/> <input type="button" value="②v"/></p> <p><input type="button" value="Add customer"/></p>				

The screenshot shows a web application interface for managing customers. The URL is `localhost:4200/user/customers`. The top navigation bar includes links for 'Products', 'Customers', 'Bills', and 'Back'. On the right, it shows 'admin admin' and a user icon. The main content area is titled 'Customers List' and contains a table with the following data:

ID	Customer name	Customer email	Bills
6	khalid	khalid@mail.com	\$ ⚒ ⚒
7	wahbi	wahbi@mail.com	\$ ⚒ ⚒

Below the table, it says '2 / 2' and '5 per page'. At the bottom left are buttons for '1' and '2'. A 'New customer' button is located at the top left of the list area.

Bills Management

Screenshot of the "localhost:4200/user/new-bill" page.

The top navigation bar shows tabs: A Front, Activité Pratique du Contrôle, Cours, Cours, Eureka, and a "+" button.

The main content area displays three customer selection boxes:

- Customer 1: Email - customer-1-new@mail.com, Select button, Selected status.
- Customer 2: Email - customer-1-new@mail.com, Select button.
- Customer 3: Email - khalid@mail.com, Select button.

Below the selection boxes are pagination controls: Previous page, 1 / 2, 100 per page, and Next page.

The "Products" section lists items for selection:

- Test product**: Price - 2000 Dhs, Quantity Left - 11 Left. Selection dropdown shows value 8, Remove button.
- Product 2**: Price - 3000 Dhs, Quantity Left - 488 Left. Selection dropdown shows value 7, Remove button.
- Product 3**: Price - 900000 Dhs, Quantity Left - 45 Left. Add button.

Below the products are additional pagination controls: Previous page, 1 / 2, 100 per page, and Next page. A yellow "Add bill" button is also present.

Screenshot of the "localhost:4200/user/bills" page.

The top navigation bar shows tabs: A Front, Activité Pratique du Contrôle, Cours, Cours, Eureka, and a "+" button. It also displays user information: admin admin and a file icon.

The main content area shows a "Bills List" table:

ID	Billing date	View	Delete
1	2022-12-22T00:00:00.000+00:00		
3	2022-12-20T00:00:00.000+00:00		

Below the table are pagination controls: 1 / 1, 5 per page, and a blue "1" button.



The screenshot shows a web application interface for managing bills. At the top, there is a navigation bar with links for 'Front', 'Activité Pratique du Contrôle', 'Cours', 'Cours', 'Eureka', and user information 'admin admin'. Below the navigation bar, the main content area displays a bill summary and a detailed product items table.

Bill 3 Ordered at 20 - 12 - 2022

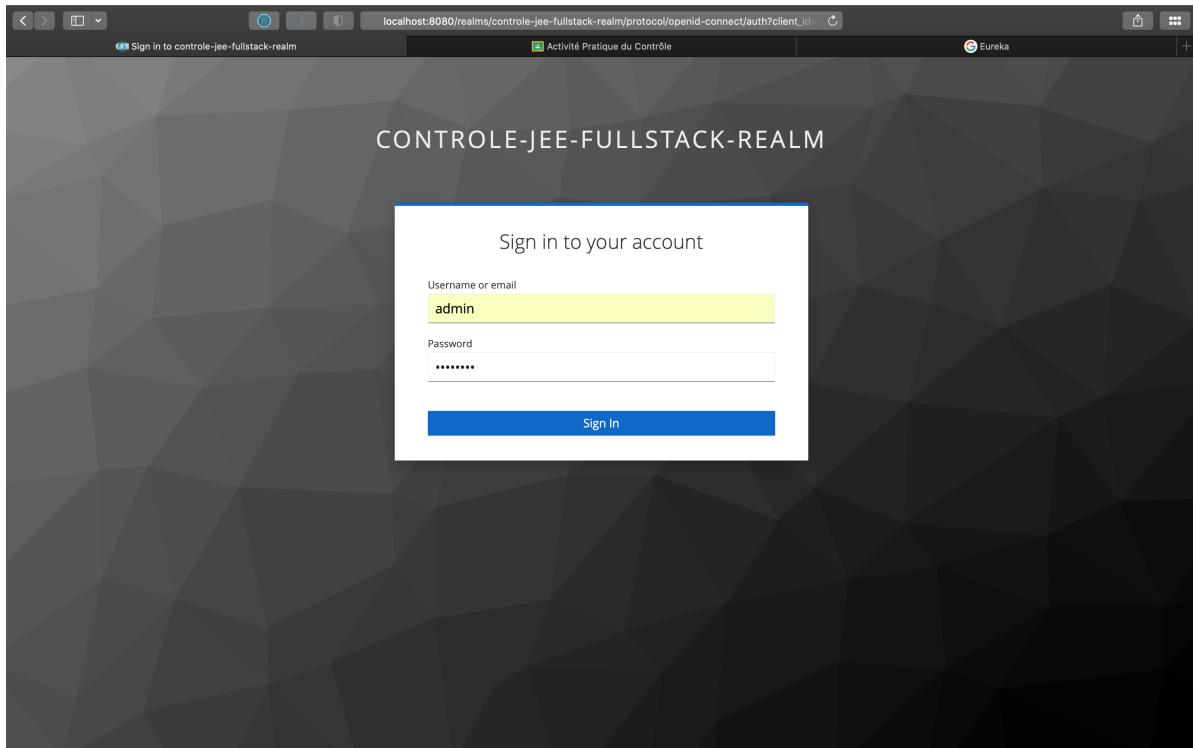
Customer wahbi wahbi@mail.com

Product items	Product id	Product name	Product unit price	Product quantity	Total price
	1	Test product	2000 Dhs	8	16000 Dhs
	2	Product 2	3000 Dhs	7	21000 Dhs

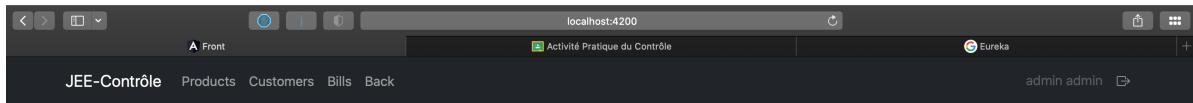


7. Keycloak Integration

Redirection vers keycloak si aucun utilisateur n'est authentifié

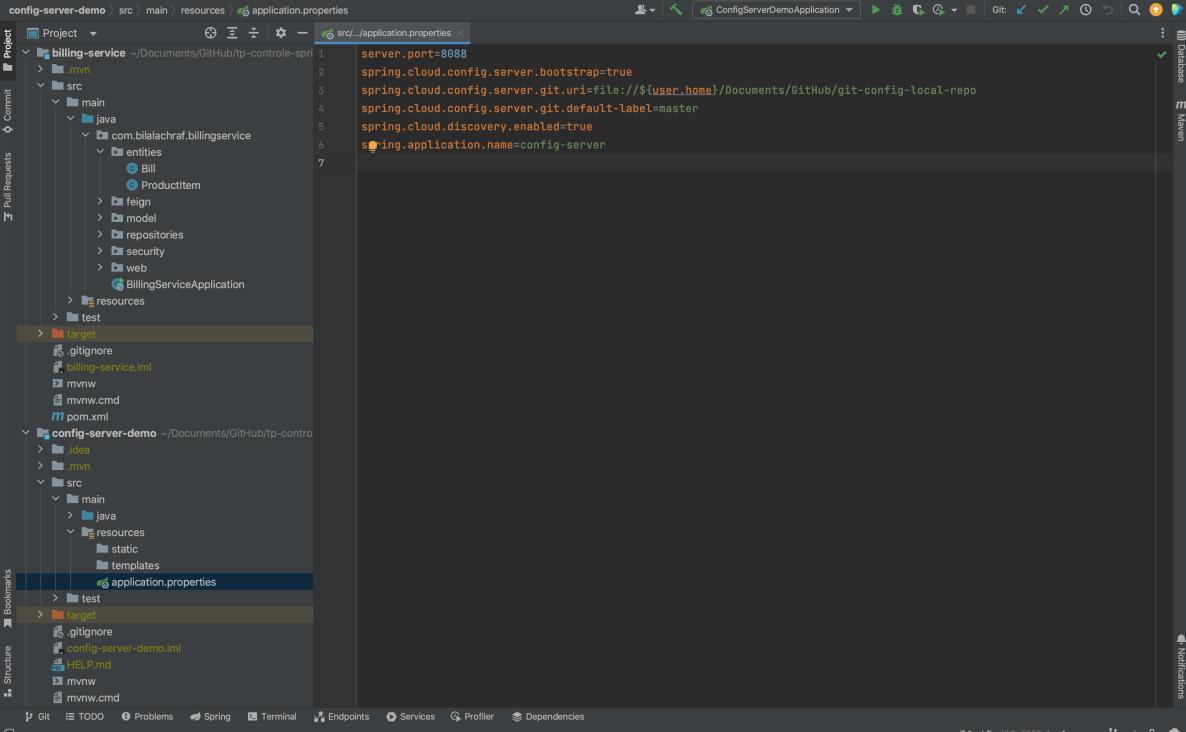


Redirection vers angular après authentification sur keycloak



8. Configuration Server

Ce service permet de centraliser les configurations de tous les autres micro services



The screenshot shows the IntelliJ IDEA interface with the following details:

- Project View:** Shows two main modules:
 - billing-service**: Contains Java code (entities, feign, model, repositories, security, web), resources, and test files.
 - config-server-demo**: Contains Java code, resources (static, templates), and test files.
- Code Editor:** Displays the `src/main/resources/application.properties` file with the following configuration:

```
server.port=8888
spring.cloud.config.server.bootstrap=true
spring.cloud.config.server.git.uri=file://${user.home}/Documents/GitHub/git-config-local-repo
spring.cloud.config.server.git.default-label=master
spring.cloud.discovery.enabled=true
spring.application.name=config-server
```
- Toolbars and Status Bar:** Standard IntelliJ IDEA toolbars and status bar at the bottom.

