

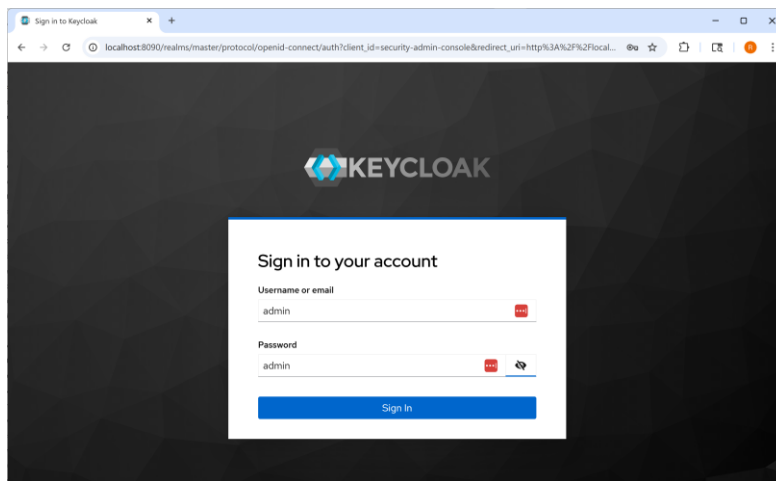
# Lab 10

## Part 1

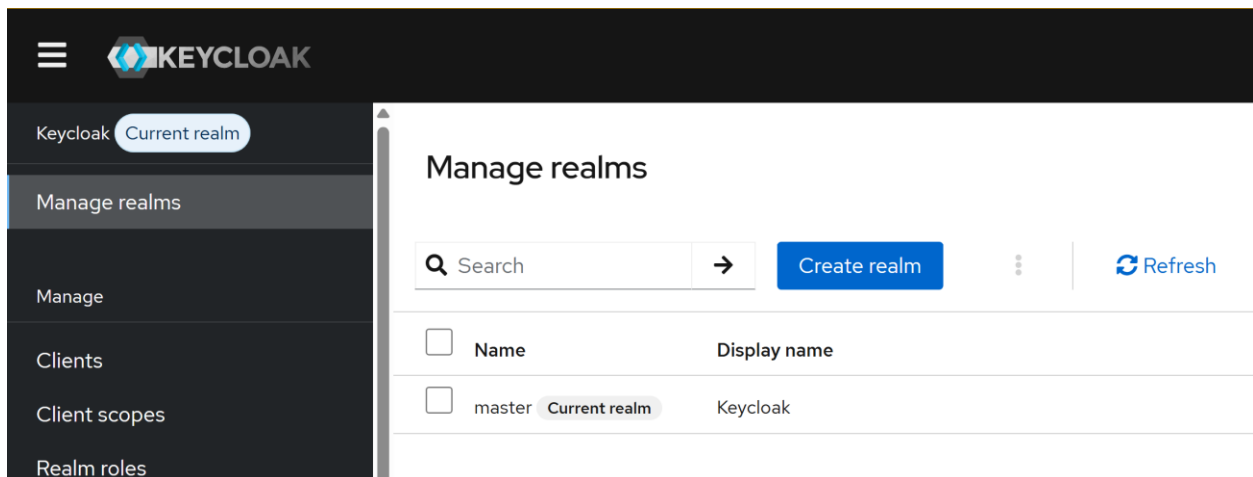
First start KeyCloak using docker:

```
docker run -d -p 8090:8080 --name keycloak -e KEYCLOAK_ADMIN=admin -e  
KEYCLOAK_ADMIN_PASSWORD=admin quay.io/keycloak/keycloak:latest start-dev
```

Then open **http://localhost:8090/**



Login with Username **admin** and password **admin**.



Select the **Manage realms** from the menu at the left, and click the **Create realm** button.

### 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

Upload a JSON file

Realm name \*

myrealm

Enabled

☒ On

Create

Cancel

Name the realm **myrealm** and click the **Create** button.

KEYCLOAK

myrealm Current realm

Manage realms

Manage

Clients

Client scopes

Realm roles

### Manage realms

Search

→

Create realm

⋮

Refresh

<input type="checkbox"/>	Name	Display name
<input type="checkbox"/>	myrealm	Current realm
<input type="checkbox"/>	master	Keycloak

The realm **myrealm** is now created is the current active realm

KEYCLOAK

myrealm Current realm

Manage realms

Manage

Clients

Client scopes

Realm roles

Users

Groups

Sessions

### Clients

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

Clients list

Initial access token

Client registration

Search for client

→

Create client

Import client

Client ID	Name	Type	Description
account	Account	OpenID Connect	—
account-console	Account Console	OpenID Connect	—
admin-cli	Admin CLI	OpenID	—

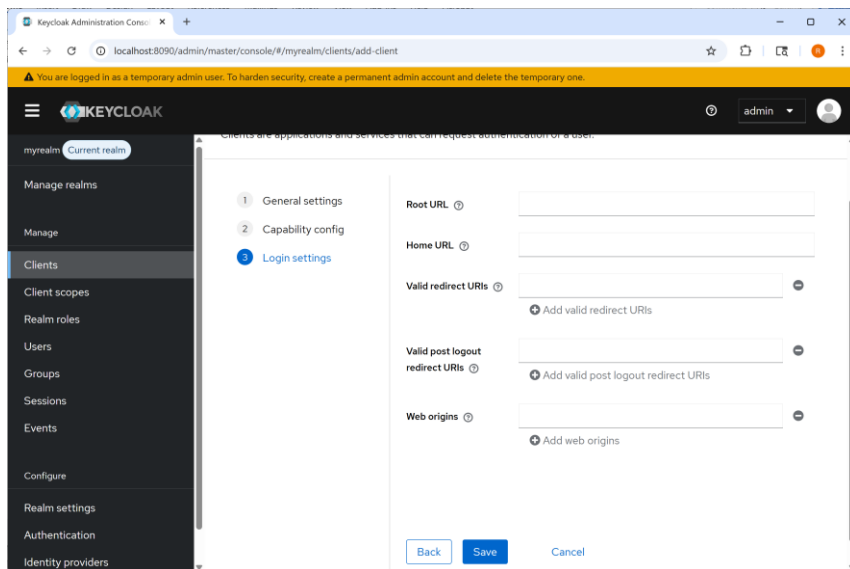
Select the **Clients** from the menu at the left, and click the **Create client** button.

The screenshot shows the Keycloak Administration Console at the URL `localhost:8090/admin/master/console/#/myrealm/clients/add-client`. A warning banner at the top states: "You are logged in as a temporary admin user. To harden security, create a permanent admin account and delete the temporary one." The left sidebar shows the navigation menu with "Clients" selected. The main content area is titled "Create client" and includes the subtext "Clients are applications and services that can request authentication of a user." The "General settings" step is active, showing the following fields: "Client type" (OpenID Connect), "Client ID" (myclient), "Name" (empty), "Description" (empty), and "Always display in UI" (Off). At the bottom are "Back", "Next", and "Cancel" buttons.

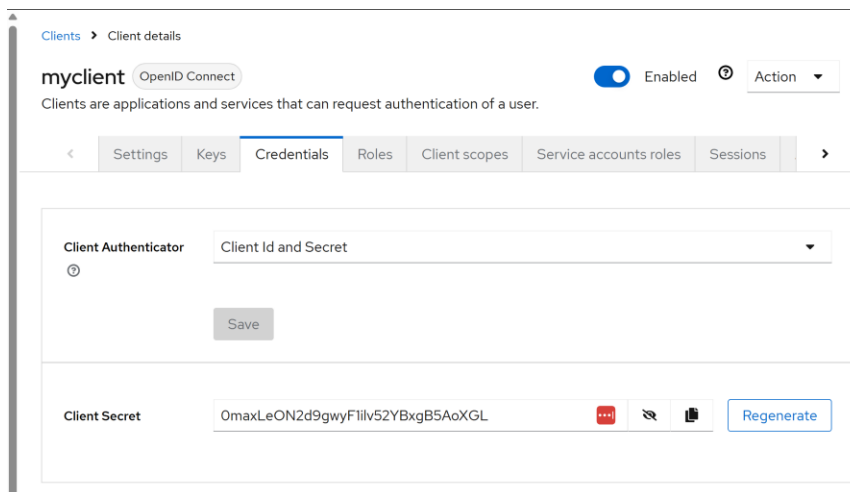
Fill in the Client ID **myclient** and click **Next**

The screenshot shows the "Create client" form at the same URL, now on the "Capability config" step. The left sidebar remains the same. The main content area shows the following configuration options: "Client authentication" (On), "Authorization" (Off), "Authentication flow" (Standard flow, Direct access grants, and Service accounts roles are checked), "Implicit flow" (unchecked), "Standard Token Exchange" (unchecked), "OAuth 2.0 Device Authorization Grant" (unchecked), "OIDC CIBA Grant" (unchecked), "PKCE Method" (Choose...), and "Require DPoP bound tokens" (Off). At the bottom are "Back", "Next", and "Cancel" buttons.

Set **Client authentication** to **On** and select **Direct access grants** and **Service accounts roles**. Then click **Next**



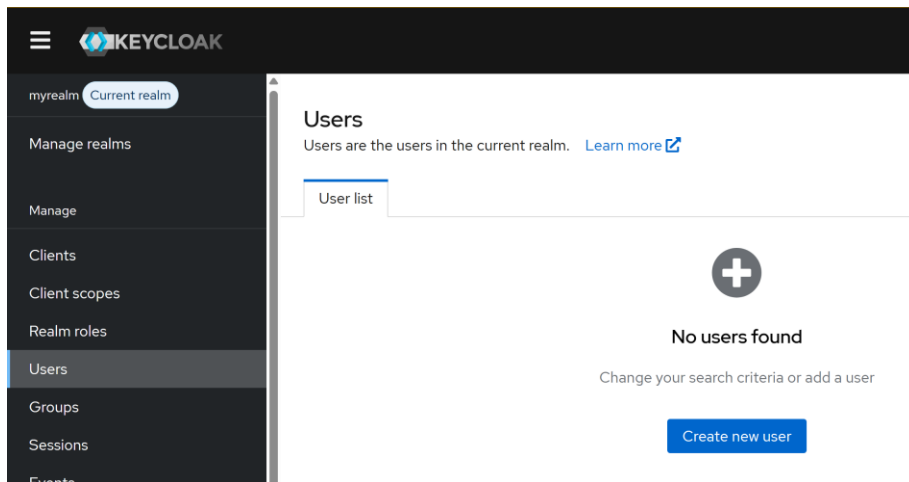
Click **Save**



Select the **Credentials** tab and copy the **Client Secret**. Save this client secret somewhere (notepad).

We have now created a client with the Client ID **myclient** and a Client secret.

Now we are going to create Users.



On the menu on the left select **Users** and click the **Create new user** button.

The screenshot shows the 'Create user' form in Keycloak. The left sidebar is the same as in the previous image. The main area is titled 'Create user' with a breadcrumb 'Users > Create user'. Below the title is a 'Required user actions' section with a 'Select action' dropdown. The 'Email verified' toggle is turned on. The 'General' section contains four text input fields: 'Username' (filled with 'john'), 'Email' (filled with 'john@gmail.com'), 'First name' (filled with 'john'), and 'Last name' (filled with 'doe'). There is a 'Join Groups' button and a 'Groups' section. At the bottom are 'Create' and 'Cancel' buttons. A 'Jump to section' dropdown is on the right, currently set to 'General'.

Fill in the **Username**, **Email**, **First name**, **Last name** and select the **Email verified** option. Then click **Create**

The screenshot shows the 'User details' page for a user named 'john' in the 'myrealm' realm. The left sidebar contains navigation links: Manage realms, Manage, Clients, Client scopes, Realm roles, Users (selected), Groups, Sessions, Events, Configure, Realm settings, Authentication, Identity providers, and User federation. The main content area has a breadcrumb 'Users > User details' and a status 'Enabled' with an 'Action' button. Below this are tabs: Details (selected), Credentials, Role mapping, Groups, Consents, Identity provider links, Sessions, and Events. The 'Details' tab shows fields for ID (e1154adc-0b0b-4054-abe2-9f7363d3ab65), Created at (11/7/2025, 3:32:12 PM), Required user actions (Select action), and Email verified (On). A 'General' section shows Username (john) and Email (john@gmail.com). At the bottom are 'Save' and 'Revert' buttons.

User **john** is created.

The screenshot shows the 'User details' page for 'john' with the 'Credentials' tab selected. It displays a large plus icon and the text 'No credentials' followed by 'This user does not have any credentials. You can set password for this user.' and a 'Set password' button.

Select the **Credentials** tab and click the **Set password** button

The screenshot shows a modal dialog titled 'Set password for john'. It contains two password input fields: 'Password' and 'Password confirmation', both with the value 'john'. There are toggle icons for each field. Below these is a 'Temporary' toggle switch set to 'Off'. At the bottom are 'Save' and 'Cancel' buttons.

Fill in the password **john** and set **Temporary** to **Off**. Click the **Save** button.

The screenshot shows the 'User details' page for a user named 'john'. The user is 'Enabled'. Below the user name, there are tabs for 'Details', 'Credentials', 'Role mapping', 'Groups', 'Consents', 'Identity provider links', 'Sessions', and 'Events'. The 'Credentials' tab is selected, showing a table with one credential:

Type	User label	Created at	Data
Password	My password	11/7/2025, 3:37:08 PM	<a href="#">Show data</a> <a href="#">Reset password</a>

User john has now password john.

Now we need to create 2 roles: **user** and **admin**.

The screenshot shows the 'Clients' page in Keycloak. The 'myclient' client is selected, and the 'Roles' tab is active. The client is 'Enabled'. Below the client name, there is a message: 'No roles for this client'. Below this message, there is a button labeled 'Create role'.

On the menu on the left, select **Clients** and select the roles tab for client **myclient**. click the **Create role** button.

The screenshot shows the 'Create role' form in the Keycloak admin console. The left sidebar contains a menu with 'myrealm' and 'Current realm' at the top, followed by 'Manage realms', 'Manage', 'Clients' (highlighted), 'Client scopes', 'Realm roles', 'Users', 'Groups', and 'Sessions'. The main content area has a breadcrumb 'Clients > Client details > Create role' and a title 'Create role'. There are two input fields: 'Role name' with the value 'USER' and 'Description' which is empty. At the bottom right are 'Save' and 'Cancel' buttons.

Make a **USER** role and click **Save**

The screenshot shows the 'Roles' tab in the Keycloak admin console for 'myclient'. The breadcrumb is 'Clients > Client details'. Below the client name 'myclient' (with 'OpenID Connect' and a toggle switch) is the text 'Clients are applications and services that can request authentication of a user.' Below this are tabs: 'Settings', 'Keys', 'Credentials', 'Roles' (active), 'Client scopes', and 'Service accounts roles'. A search bar 'Search role by name' with a magnifying glass icon and a right arrow is next to a 'Create role' button and a 'Refresh' button. Below is a table with three columns: 'Role name', 'Composite', and 'Description'.

Role name	Composite	Description
<a href="#">ADMIN</a>	False	—
<a href="#">USER</a>	False	—

In the same way create an **ADMIN** role.

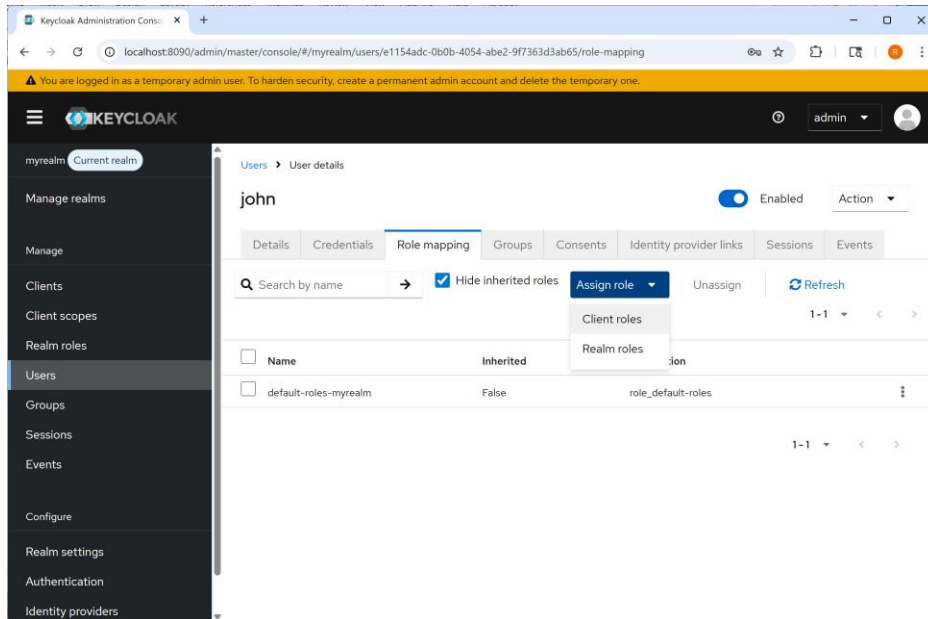
Now go back to the Users on the left menu and go to user john.

The screenshot shows the 'Role mapping' tab in the Keycloak admin console for user 'john'. The breadcrumb is 'Users > User details'. The user name 'john' is at the top left, followed by an 'Enabled' toggle switch and an 'Action' dropdown. Below are tabs: 'Details', 'Credentials', 'Role mapping' (active), 'Groups', 'Consents', 'Identity provider links', 'Sessions', and 'Events'. A search bar 'Search by name' with a magnifying glass icon and a right arrow is next to a 'Hide inherited roles' checkbox (checked), an 'Assign role' dropdown, an 'Unassign' button, and a 'Refresh' button. Below is a table with three columns: 'Name', 'Inherited', and 'Description'.

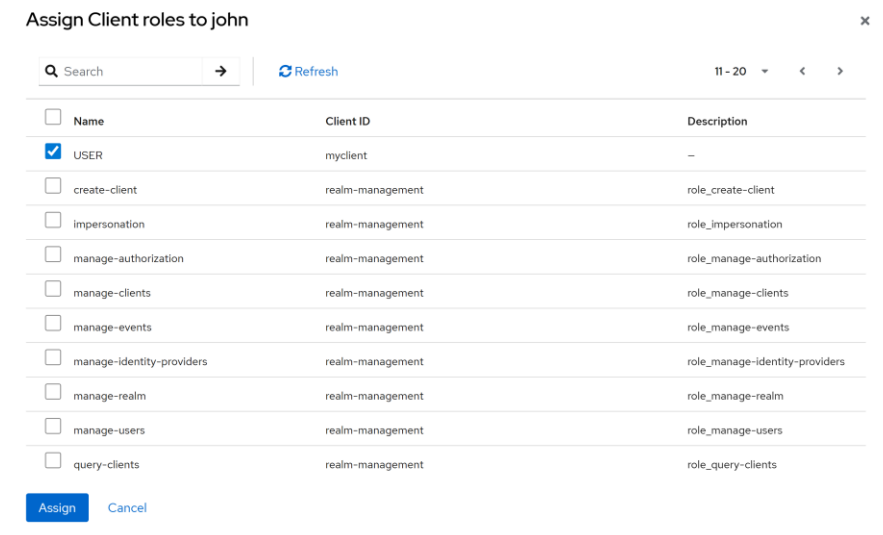
Name	Inherited	Description
<input type="checkbox"/> default-roles-myrealm	False	role_default-roles



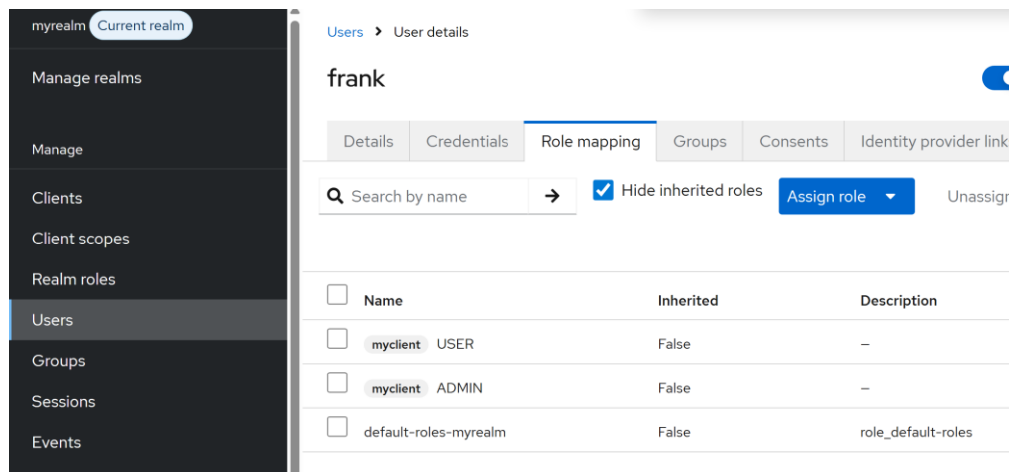
Select the **Role mapping** tab and click the **Assign role** pull down list.



Select **Client roles**.

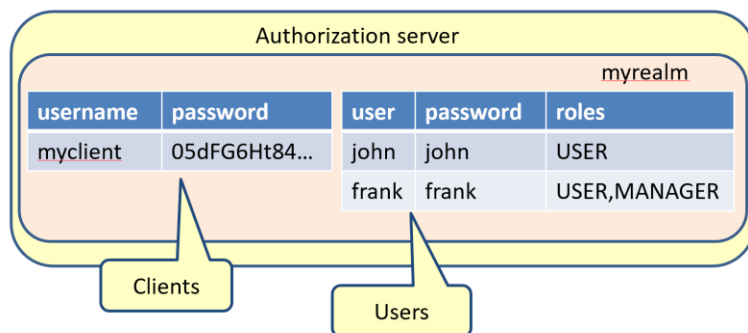


Select the **USER** role and click **Assign**.



In the same way create user **frank** with password **frank** and roles **USER** and **ADMIN**

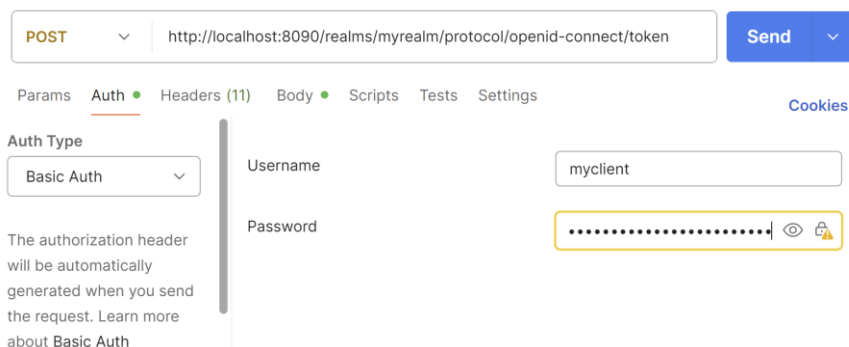
We now have the following configuration in KeyCloak:



Now we are going to retrieve the token for users john and frank.

Open **Postman** and create a **POST** request for the URL

**http://localhost:8090/realms/myrealm/protocol/openid-connect/token**



In the **Auth** tab, select **Basic Auth** and fill in the **username** and **secret** of the **Client** configured in KeyCloak.



Now check the token for user **john**

```

{
  "resource_access": {
    "myclient": {
      "roles": [
        "USER"
      ]
    },
    "account": {
      "roles": [
        "manage-account",
        "manage-account-links",
        "view-profile"
      ]
    }
  },
  "scope": "openid email profile",
  "email_verified": true,
  "name": "john doe",
  "preferred_username": "john",
  "given_name": "john",
  "family_name": "doe",
  "email": "john@gmail.com"
}

```

Our KeyCloak authorization server is now working.

Now you are ready to run the given code for this lecture. Run the given applications and check if they work correctly.

## **Part 2**

Now change the in-memory users so that we have 3 users:

One with the role customer, one with the roles customer and employee, and one with the roles customer, employee and manager.

Then create 3 microservices A, B and C.

C contains salary data

B contains employee contact data (phone)

A is our actual company service that is used by customers, employees and managers

In A you can call productdata that is accessible by all customers, employees and managers

In A you can call employee contact data that is accessible only by employees and managers. A will call B to get the actual employee contact data.

In A you can call salary data that is accessible only by managers. A will call C to get the actual salary data.

Check that everyone can call product data

Check that only employees and managers can call employee data

Check that only managers can call salary data