Inhaltsverzeichnis

1 Setup Keycloak Server

1. Download the latest keycloak server from https://www.keycloak.org/downloads.
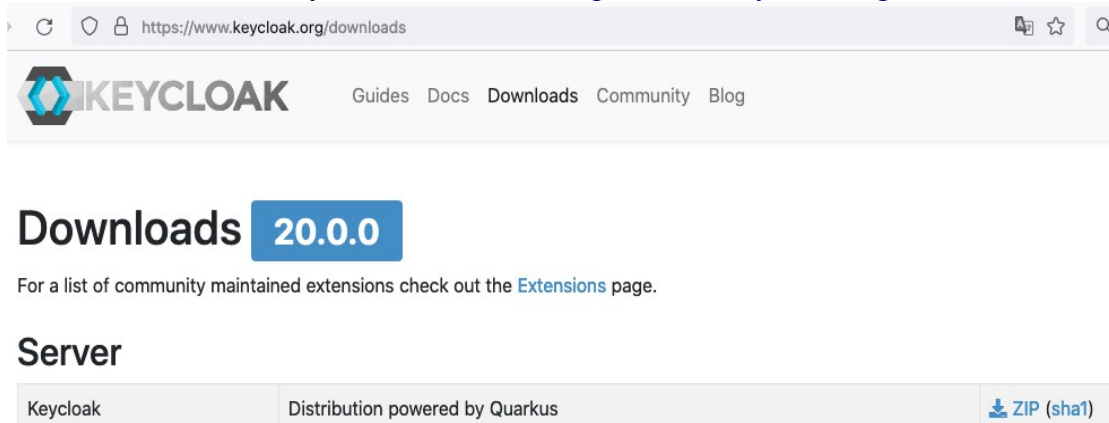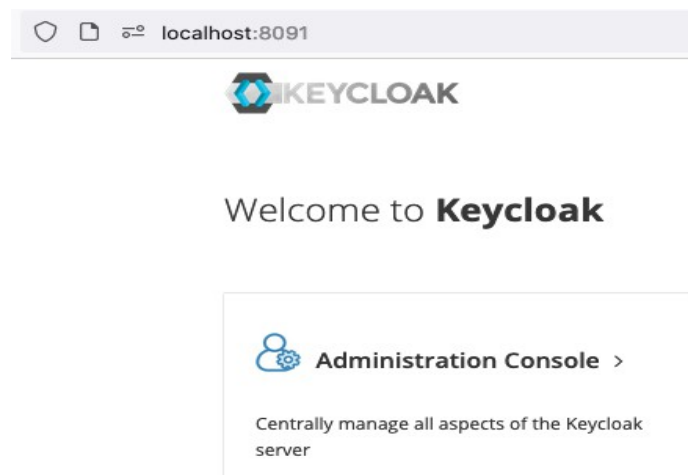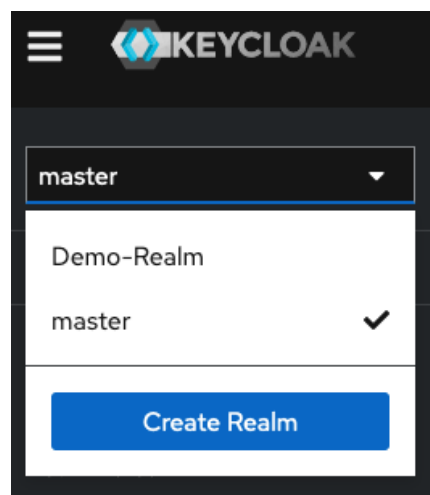

2. Unzip, open a terminal and change to directory keycloak-20.0.0 an run: *bin/kc.sh start-dev --http-port 8091*

3. Open keycloak admin console
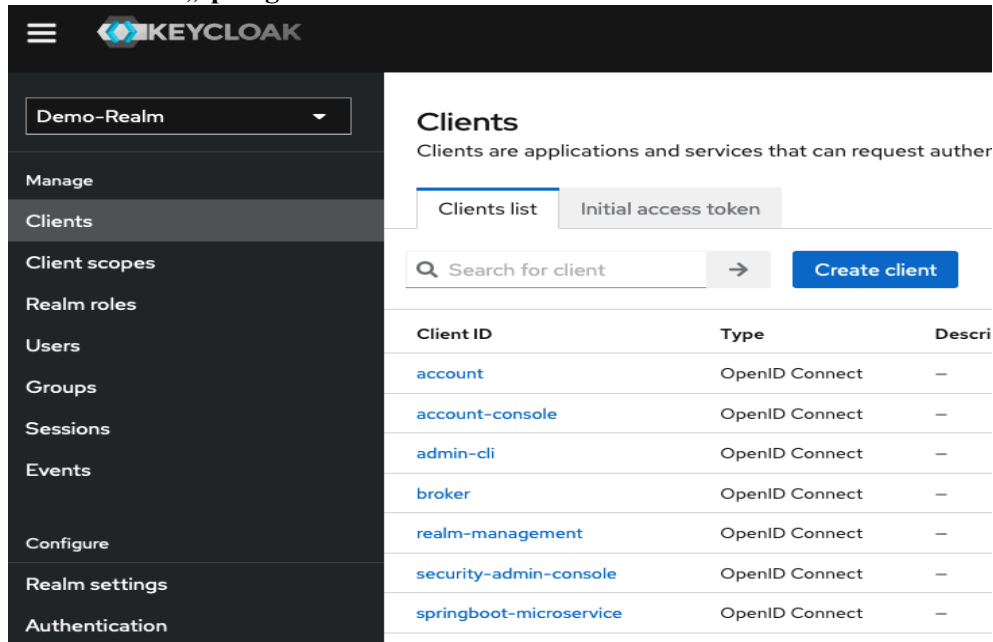

4. Have as look at https://www.djamware.com/post/6225b66ba88c55c95abca0b6/spring-boot-security-postgresql-and-keycloak-rest-api-oauth2
5. Create an admin account
6. Create a new Realm **Demo-Realm** and select it

7. Create a new client „**springboot-microservice**“

8. Fill in Root-Url, Home-Url, Valid-Redirect-Urls and Web-Origins and switch on „Client authenication“ and „Authorization“

9. Create two Realm-Roles **app-admin** and **app-user**

10. Switch to client **springboot-microservice** and create two roles **admin** and **user**

11. Switch to Realm-Roles and select Role **app-user**, click on Action and select „**Add associated roles**"

12. Select „filter by clients" and enter „springboot-microservices" and select role **user**

Assign roles to app-user account ✕

	Name		Description
☐	springboot-microservice	admin	
☐	springboot-microservice	uma_protection	
☑	springboot-microservice	user	

🔽 Filter by clients ▾   🔍 spring   ✕ →   1 - 3 ▾ ‹ ›

1 - 3 ▾ ‹ ›

Assign   Cancel

13. Realm-Role app-user is now a composite role

Realm roles › Role details

app-user Composite

Details	**Associated roles**	Attributes	Users in role	Permissions

🔍 Search by name → ☑ Hide inherited roles **Assign role** Unassign

	Name	Inherited	Description
☐	springboot-microservice user	False	–

14. Do the same with Realm-Role app-admin and associate role admin

Realm roles › Role details

app-admin Composite

Details	**Associated roles**	Attributes	Users in role	Permissions

🔍 Search by name → ☑ Hide inherited roles **Assign role** Unassign

	Name	Inherited
☐	springboot-microservice admin	False

15. Create 3 Users named employee1 – employee3, set „Email verified" to „On"

Create user

Username *	employee1
Email	m0.m0@company.com
Email verified ⓘ	🔵 On
First name	Max0
Last name	Mustermann0
Required user actions ⓘ	Select action
Groups ⓘ	Join Groups

Create   Cancel

16. Set user credentials

employee1

Details	Attributes	Credentials	Role mapping	Gr

ⓘ	Type	User label
⠿	Password	My password 🖉

17. Assign Realm-Role **app-user** to employee1

employee1

Details	Attributes	Credentials	**Role mapping**	Groups	Consents	Identity provide

Q Search by name → ☑ Hide inherited roles **Assign role** Unassign

	Name	Inherited	Description
☐	app-user	False	–
☐	default-roles-demo-realm	False	${role_default-roles}

18. Remove all actions in field „Required user action" and save
19. Repeat everything from point 14-17 with employee2 and assign Realm-Role app-admin

employee2

Details	Attributes	Credentials	**Role mapping**	Groups	Con

Q Search by name → ☑ Hide inherited roles **Assign role**

	Name	Inherited
☐	default-roles-demo-realm	False
☐	app-admin	False

20. Repeat everything from point 14-17 with employee3 and assign Realm-Role **app-admin** and **app-user**

employee3

Details	Attributes	Credentials	**Role mapping**	Groups	Conse

Q Search by name → ☑ Hide inherited roles **Assign role**

	Name	Inherited
☐	app-user	False
☐	default-roles-demo-realm	False
☐	app-admin	False

21. Switch to client scopes and add new scope read

Client scopes

Client scopes are a common set of protocol mappers and roles that are shared between

	Name	Assigned type	Protocol
☐	read	None ▼	OpenID Conr

22. Assign Realm-Roles app-admin, app-user and Roles user and admin to scope read

Client scopes > Client scope details

read openid-connect

| Settings | Mappers | **Scope** |

🔍 Search by name → ☑ Hide inherited roles   **Assign role**

	Name	Inherited
☐	app-user	False
☐	app-admin	False
☐	springboot-microservice admin	False
☐	springboot-microservice user	False

23. Switch to client „**springboot-microservices**" and add scope read as „**Optional**"

Clients > Client details

springboot-microservice OpenID Connect

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

| Settings | Keys | Credentials | Roles | **Client scopes** | Authorization | Service |

Setup | Evaluate

▼ Name ▼   🔍 read   ✕ →   **Add client scope**

	Assigned client scope	Assigned type	Descri
☐	springboot-microservice-dedicated	none ▼	Dedica
☐	read	Optional ▼	

24. Switch to Realm settings an open „OpenID Endpoint Configuration"
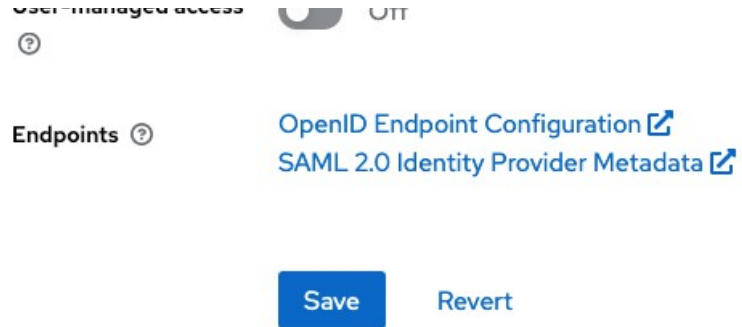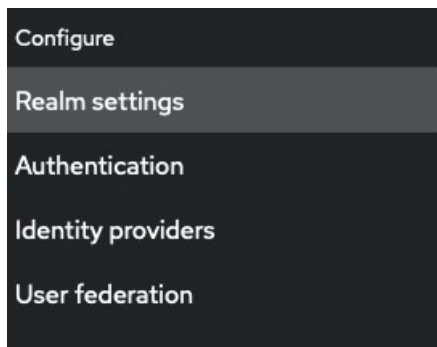
26. Test token endpoint → Copy client secret from Client „springboot-microservices"
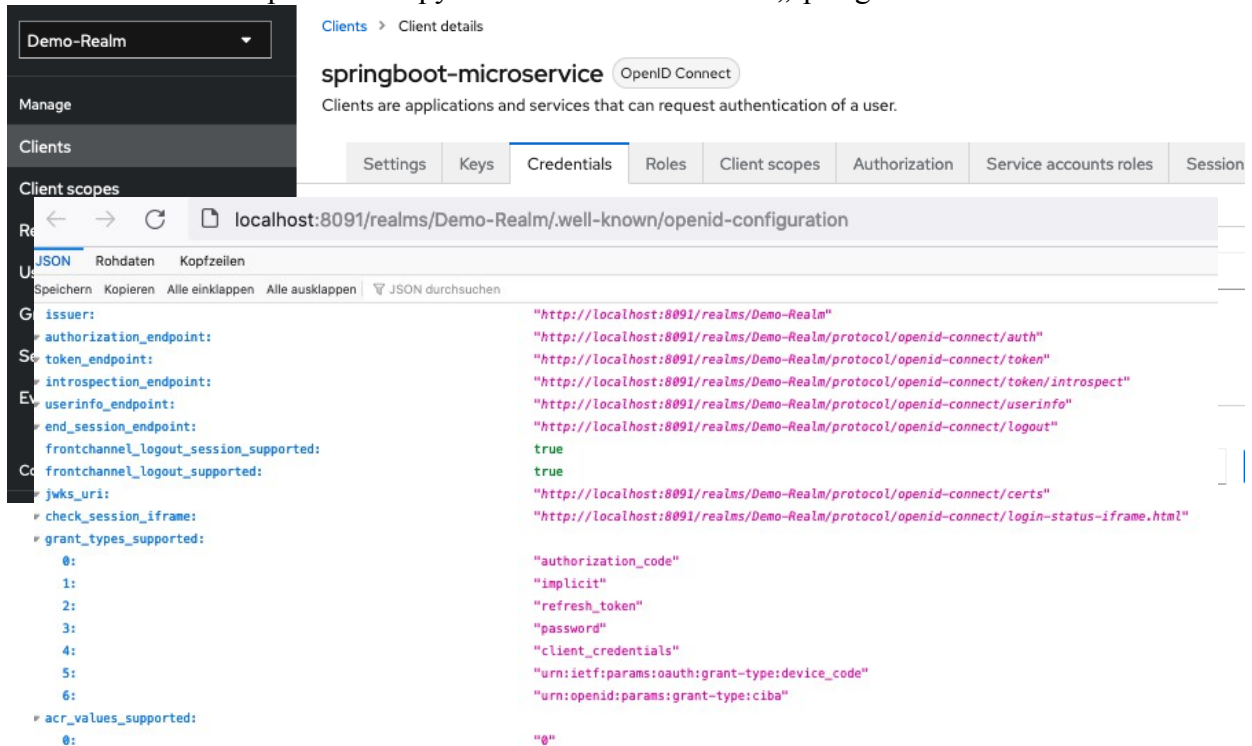
27. Use Postman or a REST client of your choice to receive an access token

Request ⤢ ⚙ +

| POST ▾ | http://localhost:8091/realms/Demo-Realm/protocol/openid-connect/token | **Send request** |

Headers ›
Basic auth ›
Request body ⌄

| Type | URLencoded form data ▾ |

client_id	springboot-microservice	🗑
grant_type	password	🗑
client_secret	*Your client secret*	🗑
username	employee1	🗑
password	*Your user password*	🗑
scope	profile roles email read	🗑

+Add parameter

Response (0.077s) - http://localhost:8091/realms/Demo-Realm/protocol/openid-connect/token

200 OK

Headers ›

{
  "access_token": "eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJnS0g5ZjdUUGRucnVncHFuNGRPV0x5X202X2drZ0FWU2RtZ0hWMFlfVFFNIn0.eyJleHAiOjE2Njc3Mzc10DYsImlhdCI6MTY2NzczNzI4NiwianRpIjoiODZiZjU1ZDYtMTk0Yy00NTJkLTk5N2YtNjY4YTcyMTU4ZTdhIiwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdDo4MDkxL3JlYWxtcy9EZW1vLVJlYWxtIiwiYXVkIjoiYWNjb3VudCIsInN1YiI6IjMyM2I3NDJiLTc0ODMtNGZjYi1hYzRhLTAxNTYzMzI5NmM3OSIsInR5cCI6IkJlYXJlciIsImF6cCI6InNwcmluZ2Jvb3QtbWljcm9zZXJ2aWNlIiwic2Vzc2lvbl9zdGF0ZSI6IjEwNzk1MTRiLTQ5OWItNDY2Zi1hMDVhLTRhNTQxZjM3NjRjOSIsImFjciI6IjEiLCJhbGxvd2VkLW9yaWdpbnMiOlsiaHR0cDovL2xvY2FsaG9zdDo4MDgwIl0sInJlYWxtX2FjY2VzcyI6eyJyb2xlcyI6WyJvZmZsaW5lX2FjY2VzcyIsImRlZmF1bHQtcm9sZXMtZGVtby1yZWFsbSIsInVtYV9hdXRob3JpemF0aW9uIiwiYXBwLXVzZXIiLCJkZWZhdWx0LXVzZXIiXX0sInJlc291cmNlX2FjY2VzcyI6eyJhY2NvdW50Ijp7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50IiwibWFuYWdlLWFjY291bnQtbGlua3MiLCJ2aWV3LXByb2ZpbGUiXX19LCJzY29wZSI6InByb2ZpbGUgcmVhZCBlbWFpbCIsInNpZCI6IjEwNzk1MTRiLTQ5OWItNDY2Zi1hMDVhLTRhNTQxZjM3NjRjOSIsImVtYWlsX3ZlcmlmaWVkIjp0cnVlLCJuYW1lIjoiR2lsYmVydCBIb2ZmZ2dMOkdHRlciIsInByZWZlcnJlZF91c2VybmFtZSI6ImVtcGxveWVlMSIsImdpdmVuX25hbWUiOiJHaWxiZXJ0IiwiZmFtaWx5X25hbWUiOiJIb2ZmZdGMOkdHRlciIsImVtYWlsIjoiZ2lsYmVydC5ob2ZmZGFldHRlcjFAZ21haWwuY29tIn0.nXR9oAV30cZeLyaasKlCwEFGVt7koUl0-fLtoC1OBqK1kf7O4fqNEbW10r34XX9hUMrR2oFMAtMiSbLA0VKs5HfEgpE2sFmUq6KIKRAb0XnkxqTH2yrR4j0vNBW-xGd_d2W2cXhmGe3zhgDYC9UIspvSEAUVt_4JyuSggSitWOcg2x_00aDyrNXVkYaMLfRc-5SxIpbP9SPuDF9hGrRqILs_xawecLIZzbeu13rNwclYb896x2kXnRFeW2WA501brDLgNKmihaoUNDPIyXm2-K9wMnzW2FVFy84oM1qBS_-sGvTu-6elAuSOYkuXyYqUpdUGyUupZh2P9d5THLYdpw",

28. Use Refresh token to receive a new access token

| POST ▾ | http://localhost:8091/realms/Demo-Realm/protocol/openid-connect/token | **Send request** |

Headers ›
Basic auth ›
Request body ⌄

| Type | URLencoded form data ▾ |

client_id	springboot-microservice	🗑
grant_type	refresh_token	🗑
client_secret	*Your client secret*	🗑
refresh_token	NGM5In0.HlDHATdRBcxc5GrB7zrEnTtYr2XbZIpVGMyqqT0ot1E	🗑

+Add parameter

Response (0.011s) - http://localhost:8091/realms/Demo-Realm/protocol/openid-connect/token

200 OK

Headers ›

{
  "access_token": "eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJnS0g5ZjdUUGRucnVncHFuNGRPV0x5X202X2drZ0FWU2RtZ0hWMFlfVFFNIn0.eyJleHAiOjE2Njc3Mzc3NzEsImlhdCI6MTY2NzczNzQ3MSwianRpIjoiN2U1NmY1MzItNjc2Mi00ZGQ3LThlZTItZDg3MDMwZGY5N2M3IiwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdDo4MDkxL3JlYWxtcy9EZW1vLVJlYWxtIiwiYXVkIjoiYWNjb3VudCIsInN1YiI6IjMyM2I3NDJiLTc0ODMtNGZjYi1hYzRhLTAxNTYzMzI5NmM3OSIsInR5cCI6IkJlYXJlciIsImF6cCI6InNwcmluZ2Jvb3QtbWljcm9zZXJ2aWNlIiwic2Vzc2lvbl9zdGF0ZSI6IjEwNzk1MTRiLTQ5OWItNDY2Zi1hMDVhLTRhNTQxZjM3NjRjOSIsImFjciI6IjEiLCJhbGxvd2VkLW9yaWdpbnMiOlsiaHR0cDovL2xvY2FsaG9zdDo4MDgwIl0sInJlYWxtX2FjY2VzcyI6eyJyb2xlcyI6WyJvZmZsaW5lX2FjY2VzcyIsImRlZmF1bHQtcm9sZXMtZGVtby1yZWFsbSIsInVtYV9hdXRob3JpemF0aW9uIiwiYXBwLXVzZXIiLCJkZWZhdWx0LXVzZXIiXX0sInJlc291cmNlX2FjY2VzcyI6eyJhY2NvdW50Ijp7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50IiwibWFuYWdlLWFjY291bnQtbGlua3MiLCJ2aWV3LXByb2ZpbGUiXX19LCJzY29wZSI6InByb2ZpbGUgcmVhZCBlbWFpbCIsInNpZCI6IjEwNzk1MTRiLTQ5OWItNDY2Zi1hMDVhLTRhNTQxZjM3NjRjOSIsImVtYWlsX3ZlcmlmaWVkIjp0cnVlLCJuYW1lIjoiR2lsYmVydCBIb2ZmZdGMOkdHRlciIsInByZWZlcnJlZF91c2VybmFtZSI6ImVtcGxveWVlMSIsImdpdmVuX25hbWUiOiJHaWxiZXJ0IiwiZmFtaWx5X25hbWUiOiJIb2ZmZdGMOkdHRlciIsImVtYWlsIjoiZ2lsYmVydC5ob2ZmZGFldHRlcjFAZ21haWwuY29tIn0.wlYVeiuH9WIrutlYaiIBNDB05wj6oR9He5PfmiyaZEQ1cxvc69jE5WURZAZf3t2fL5nDsDOvrru9fmcY0wOemqE6274tb9b_rzOhjWb3jbmPc0kX8BgdMXhDtsRiExOOCRhIq5oB-9BKEbzjiJNjLSym8AiB7vy6ouGQ0S9MsBhHmTutHMRFZ0naNquMBVVfZ6XB_yIm3G1pmlkdY5KAttFJ_vLRpLwH_ZE3BXcssnccaRwTw_4p730ADi9daq-5sKd9teanyjePpezpXCrG2c27-RRqGHRFXNmthH8A_H2GG4jAmgdn-hQZVWV4AhjO6-1metMU3EaSkbYac5VO0Q",
  "expires_in": 300,
  "refresh_expires_in": 1800,
  "refresh_token": "eyJhbGciOiJIUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICI1OGM0MTM5Ny04MmRlLTRlNmYtYWJiYy1kM2IyM2E1ZTIzMDkifQ.eyJleHAiOjE2Njc3MzkyNzEsImlhdCI6MTY2NzczNzQ3MSwianRpIjoiYjIyN2JiNzktYThiYi00YjMyLTkwZmMtY2FkMDZkNmY5OGYwIiwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdDo4MDkxL3JlYWxtcy9EZW1vLVJlYWxtIiwiYXVkIjoiaHR0cDovL2xvY2FsaG9zdDo4MDkxL3JlYWxtcy9EZW1vLVJlYWxtIiwic3ViIjoiMzIzYjc0MmItNzQ4My00ZmNiLWFjNGEtMDE1NjMzMjk2Yzc5IiwidHlwIjoiUmVmcmVzaCIsImF6cCI6InNwcmluZ2Jvb3QtbWljcm9zZXJ2aWNlIiwic2Vzc2lvbl9zdGF0ZSI6IjEwNzk1MTRiLTQ5OWItNDY2Zi1hMDVhLTRhNTQxZjM3NjRjOSIsInNjb3BlIjoicHJvZmlsZSByZWFkIGVtYWlsIiwic2lkIjoiMTA3OTUxNGItNDk5Yi00NjZmLWEwNWEtNGE1NDFmMzc2NGM5In0.p57rtrAGpOKmxnmKKUFK173C22bCwdGAZ-BdLEomPfQ",

29. Open https://jwt.io/ to decode our access token
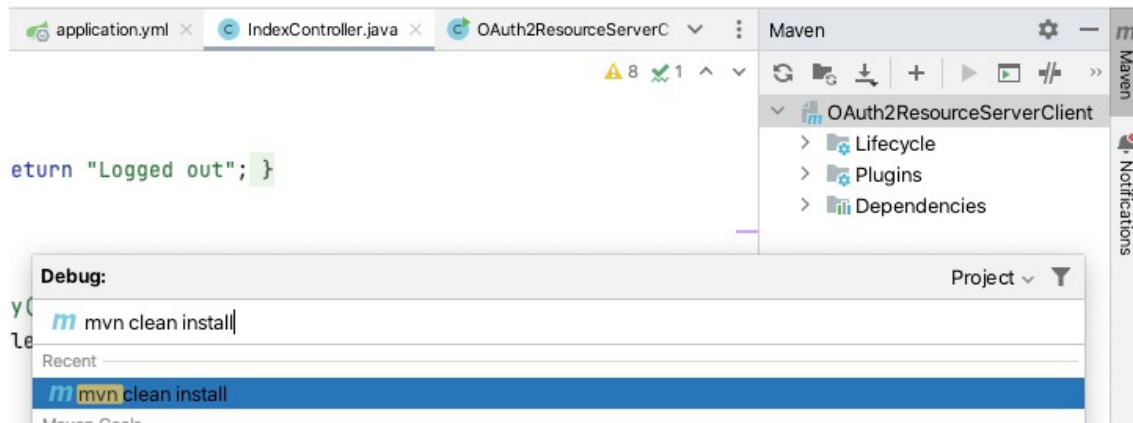30. See section resource-access for our applied user role

```
",
"realm_access": {
  "roles": [
    "offline_access",
    "default-roles-demo-realm",
    "uma_authorization",
    "app-user"
  ]
},
"resource_access": {
  "springboot-microservice": {
    "roles": [
      "user"
    ]
  },
  "account": {
```
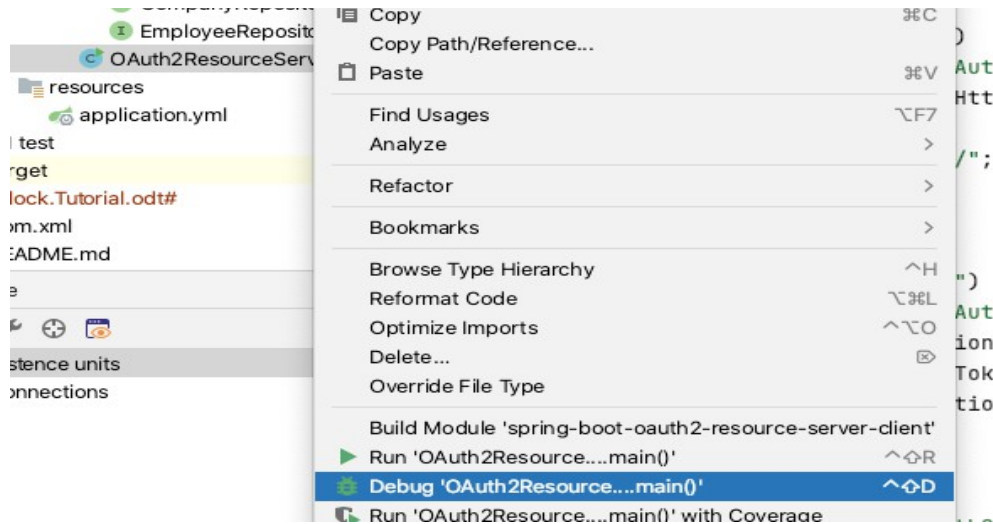
2 Setup Spring boot project

- Clone from git@github.com:GilbertHofstaetter/spring-boot-oauth2-resource-server-client.git
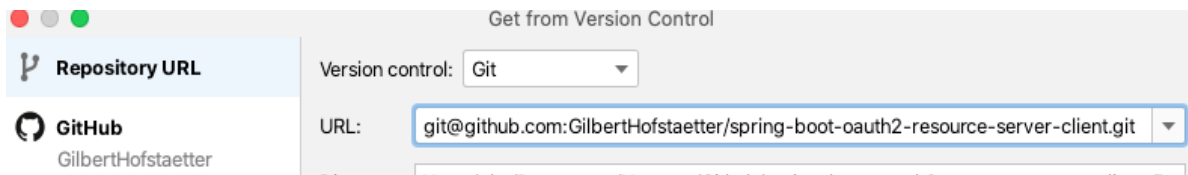
- mvn clean install

- Start project

3 Test oauth2 login

- Open http://localhost:8080/api/employee/list in your browser

- Enter login infos for user employee1 → Data gets listed → Role user is applied

- Open http://localhost:8080/api/company/list → allready authenticated

- Try to open http://localhost:8080/userOAuth2 → a blank page appears and status 403 is reported → endpoint ../userOAuth2 needs admin role to perform → logout and sign in as user employee2 or employee3
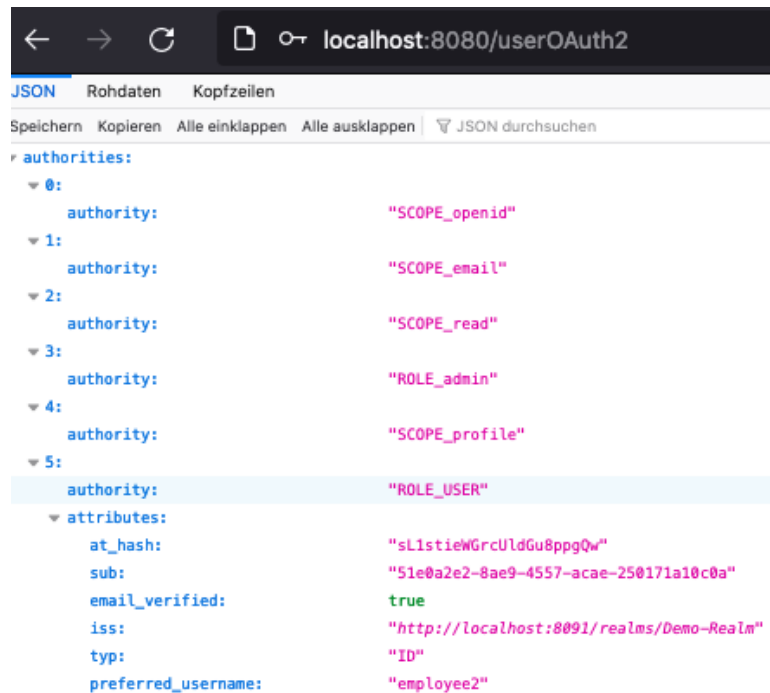

- Logout with http://localhost:8080/logout

Logged out
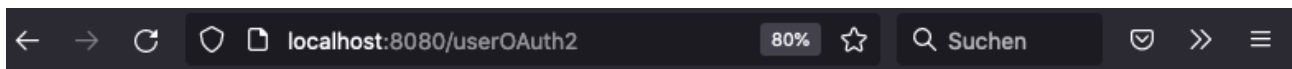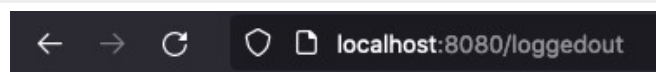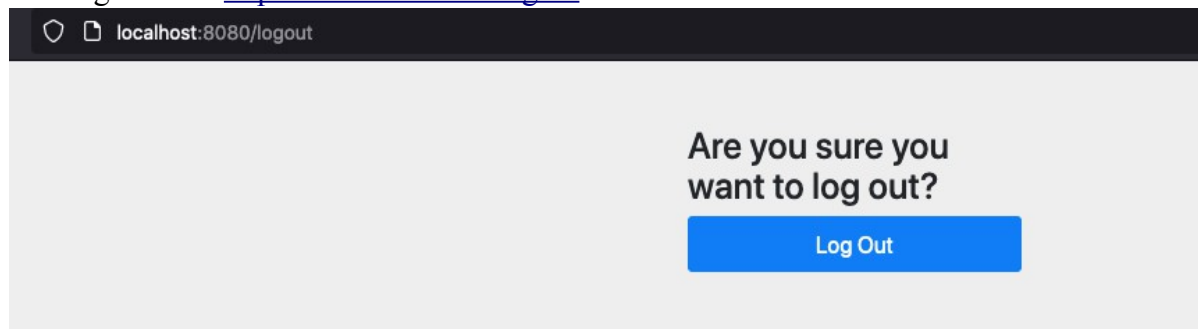
- Open again http://localhost:8080/userOAuth2 → login with user employee2 or employee3

4 Test JWT access

- Access http://localhost:8080/api/employee/list via REST client
- Retrieve an access token as employee3 (don't forget to request optional **scope read**)

{
  "access_token": "eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJnS0g5ZjdUUGRucnVncHFuNGRPV0x5X202X2drZ0FWU2RtZ0hWMFlfVFFNIn0.eyJleHAiOjE2Njc3NDAzOTEsImlhdCI6MTY2Nzc0MDA5MSwianRpIjoiYmIxMWNmNjQtN2JmNC00OY2MxLWI0YmMtNzk0MTgyNmMxYmYwIiwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdDo4MDkxL3JlYWxtcy9EZW1vLVJlYWxtIiwiYXVkIjoiYWNjb3VudCIsInN1YiI6IjJmNDA0ODhkLTU0MzMtNDI0NC1hYmEzLTdhNWE5OWQxODAwZCIsInR5cCI6IkJlYXJlciIsImF6cCI6InNwcmluZ2Jvb3Qtb..."
}

- Call http://localhost:8080/api/employee/list with Authorisation-Header Bearer *y**our access token ***

GET	http://localhost:8080/api/employee/list		Send request

Headers ⌄

Authorization	b_UzmUbsK4EFQXcqASKJbsQ0vkJNSFfaL—bQtLAKXscu0XGnhpA	🗑

+Add header

Basic auth >

Response (0.104s) - http://localhost:8080/api/employee/list

200

Headers >

```json
[
  {
    "id": 1,
    "firstName": "Max0",
    "lastName": "Mustermann0",
    "email": "m0.m0@company.com",
    "birthDate": "2022-11-06T12:37:52.319+00:00",
    "company": {
      "id": 4,
      "name": "Company"
    }
  },
  {
    "id": 2,
    "firstName": "Max1",
    "lastName": "Mustermann1",
    "email": "m1.m1@company.com",
    "birthDate": "2022-11-06T12:37:52.425+00:00",
    "company": 4
  },
  {
    "id": 3,
    "firstName": "Max2",
    "lastName": "Mustermann2",
    "email": "m2.m2@company.com",
    "birthDate": "2022-11-06T12:37:52.427+00:00",
    "company": 4
  }
]
```

- Call http://localhost:8080/api/company/list

GET	http://localhost:8080/api/company/list		Send request

Headers ⌄

Authorization	b_UzmUbsK4EFQXcqASKJbsQ0vkJNSFfaL—bQtLAKXscu0XGnhpA	🗑

+Add header

Basic auth >

Response (0.035s) - http://localhost:8080/api/company/list

200

Headers >

```json
[
  {
    "id": 4,
    "name": "Company",
    "employees": [
      {
        "id": 2,
        "firstName": "Max1",
        "lastName": "Mustermann1",
        "email": "m1.m1@company.com",
        "birthDate": "2022-11-06T12:37:52.425+00:00"
      },
      {
        "id": 3,
        "firstName": "Max2",
        "lastName": "Mustermann2",
        "email": "m2.m2@company.com",
        "birthDate": "2022-11-06T12:37:52.427+00:00"
      },
      {
        "id": 1,
        "firstName": "Max0",
        "lastName": "Mustermann0",
        "email": "m0.m0@company.com",
        "birthDate": "2022-11-06T12:37:52.319+00:00"
      }
    ]
  }
]
```

- Call http://localhost:8080/userJwt

5 Springdoc/Openapi

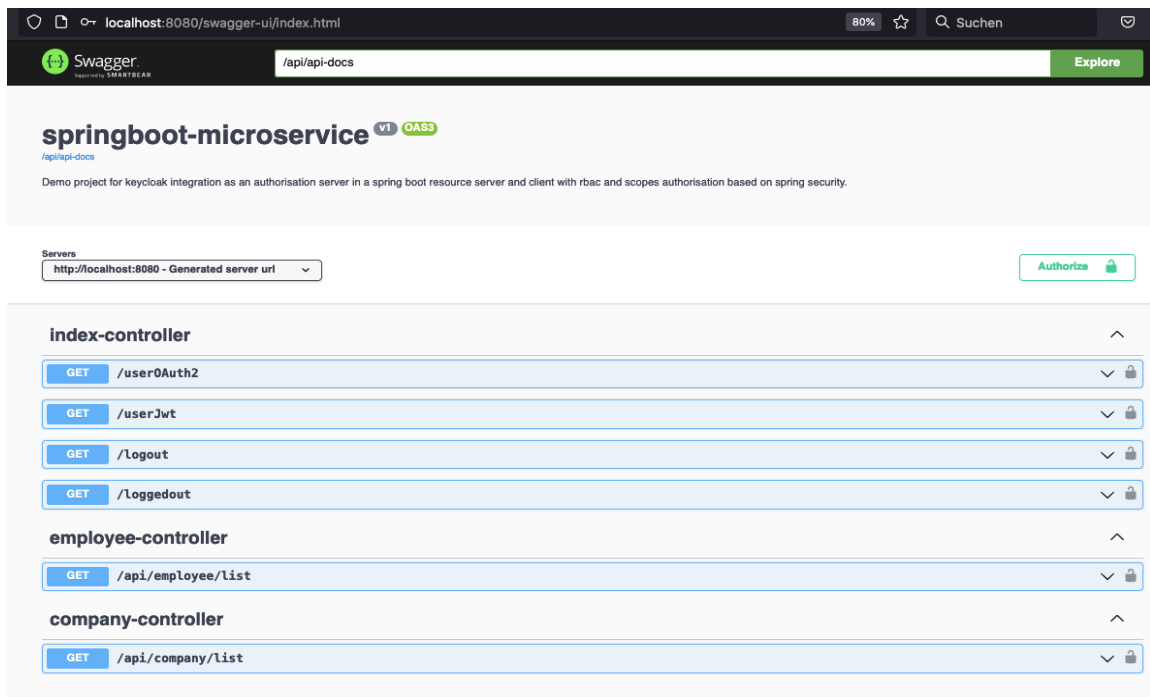- Open http://localhost:8080/swagger-ui.html in a Browser
- Login as employee2 (admin Role)

- Swagger-ui opens

- Click „Authorize"
- Fill in employee3 as user, userpassword, client-id (springboot-microservice), client secret and select scope „read"
  - Password Flow is used

Available authorizations

Scopes are used to grant an application different levels of access to data on behalf of the end user. Each API may declare one or more scopes.

API requires the following scopes. Select which ones you want to grant to Swagger UI.

keycloak (OAuth2, password)

Authorized

Token URL: http://localhost:8091/realms/Demo-Realm/protocol/openid-connect/token
Flow: password
username: employee3
password: ******
Client credentials location: basic
client_id: ******
client_secret: ******

Logout   Close

- Open Endpoint /api/employee/list
  - Bearer token has been added to Authorization Header
  - → Execute → Call successfull
  - Bearer token from Available authorizations is used instead of previous user authorization to swagger-ui

6 Vaadin front end

- Call http://localhost:8080/ui
- Keycloak login opens

- Enter User and password
- Redirect to http://localhost:8080/ui

- Open Employee

- Open Company

- Open Swagger (admin rights needed)

- Click „Logout" → redirect to Keycloak login