

A Quick Guide to Using Keycloak for Identity and Access Management

August 2, 2018 • By CoMakeIT • Product Implementation, Techies Corner



In this blog we will share how to use Keycloak for Identity and Access Management. but first here are some Frequently asked questions about Keycloak.

What is Keycloak?

Keycloak is an open source Identity and Access Management tool with features such as Single-Sign-On (SSO), Identity Brokering and Social Login, User Federation, Client Adapters, an Admin Console, and an Account Management Console.

Is Keycloak free?

Yes Keycloak is free, As Keycloak is open-source and has Apache License 2.0.

IAM (Identity Access Management)

IAM or IdM (Identity Management) is a framework used to authenticate the user identity and privileges. It checks whether the users have access to necessary files, networks and other resources that the user has requested. It also checks how and by whom the information can be accessed and modified by the management of descriptive information of users. IAM systems provide tools and some technologies to the administrators to change a user's role, keeping track on user activities etc.

Introduction



Identity Management has four main basic functions:

1. The pure identity function: Without regard to access or entitlements for identity creation, management, and deletion.
2. The user access(log-on) function: For example, to log-on to a service or services(a traditional view) the customer uses a smart card and its associated data.
3. The service function: For user and their devices a system delivers personalized, role-based, online, on-demand, multimedia(content) and presence-based services.
4. Identity Federation: To authenticate a user without knowing his/her password can be done by a system using federated identity.

Single Sign-On

Single sign-on (SSO) is a property of access control for multiple related and independent software systems where user login with single ID and password to gain access to a connected system/s without different usernames or passwords. SSO is typically accomplished on Lightweight Directory Access Protocol(LDAP) and stored LDAP databases on (directory) servers, an SSO can be achieved over IP networks using cookies but only if the sites share a common DNS parent domain. Shared authentication schemes include OAuth, OpenID, OpenID Connect and Facebook Connect. All the authentication schemes which we are using need user to log in their credentials every time they access a site or application, but we no need to get confused with SSO, in this single sign-on is enough to sign into different applications. By using SSO users can enter their credentials once but always when they are signed.

Benefits of using SSO

1. Reduce risk for 3rd party sites to access.
2. Reduce password debility from the different username and password combinations.

3. Reduce time spent for re-entering passwords for the same identity.
4. Reduce IT help desk calls for passwords, therefore, IT costs are also reduced.

Keycloak

Keycloak is an open source identity and access management solution which mainly aims at applications and services. Users can authenticate with Keycloak rather than individual applications. So, the applications don't have to deal with login forms, authenticating users and storing users. Once logged-in to Keycloak, users don't have to login again to access different applications. Same thing is applicable to sign-out. Keycloak offers everything a sophisticated user management tool needs – without having to log on repeatedly with every login and into every system-as well as system security, social logins, support for mobile apps and integration into other solutions. Keycloak have implementations to LDAP and Active Directory as well.

Password Policies

Each new realm created has no password policies associated with it while users can create as short, as long, as complex, as insecure a password, as they want. Simple settings are fine for development or learning Keycloak, but unacceptable in production environments.

OTP

Keycloak has a number of policies you can set up for your FreeOTP or Google Authenticator One-Time Password generator. There are two types:

1. TOTP(Time based OTP)
2. HOTP(Counter based OTP)

Client Certificate

- A client authentication certificate is a certificate used to authenticate clients during an SSL handshake, users who access a server by exchanging the client authentication certificate. A client certificate would typically contain pertinent information like a digital signature, expiration date, name of client, name of CA (Certificate Authority), revocation status, SSL/TLS version number, serial number, and possibly more, all structured using the X.509 standard. Very popular web browsers like Firefox, Chrome, Safari, and Internet Explorer can readily support client certificates.
- If an application is enabled with client certificate authentication, only users who attempt to connect from clients loaded with the right client certificates will succeed. Even if a legitimate user attempts to connect with the right username and password, if that user isn't on a client application loaded with the right client certificate, that user will not be granted access. In fact, if that user's connecting from a Web browser, the login page (where he's supposed to enter his username and password) might not even load at all like the one shown below.
- A server certificate is sent from the server to the client at the start of a session and is used by the client to authenticate the server. A client certificate, on the other hand, is sent from the client to the server at the start of a session and is used by the server to authenticate the client.

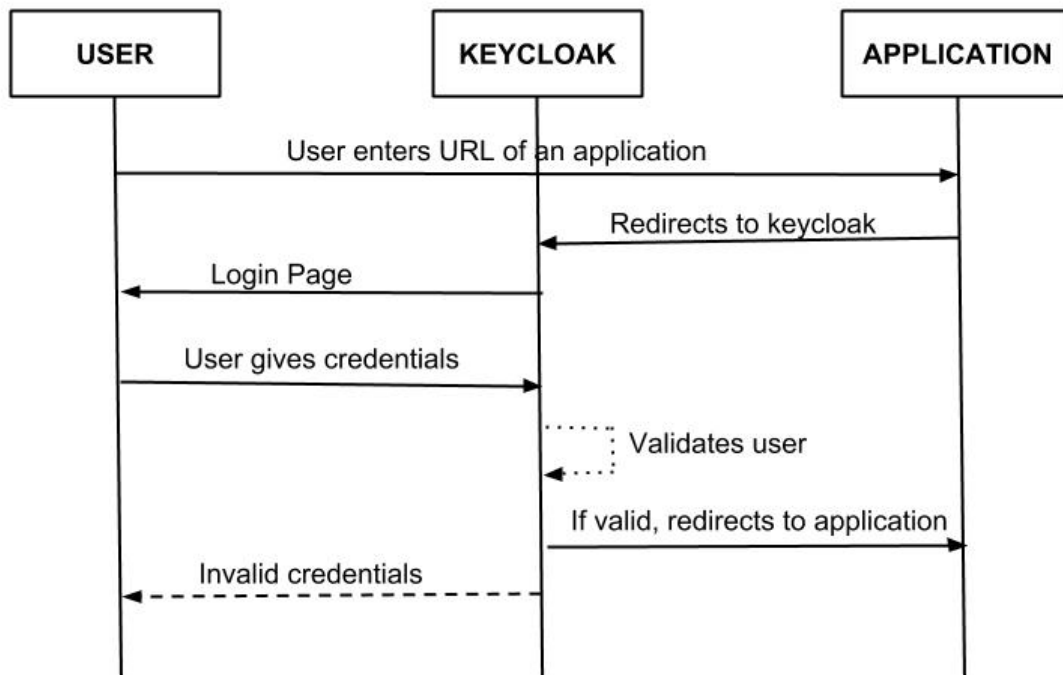
Keycloak Working Procedure

On a complete system secured with keycloak:

A user clicks from a public page to navigate to protected area within the application. The link to this protected area is in the application settings in keycloak admin console.

- The user will be redirected indeed to the keycloak authentication page. After providing username and password, keycloak redirects the user back to the application again with a code that is valid to a very short span of time.

- The application communicates this code to keycloak along with the application ID and the application secret, then keycloak replies with the Access token, ID token, and a Refresh token. Your application will need only one of these tokens to see which claims the user has, and according to the claims, the user will be granted or denied access to the requested protected URL(s).

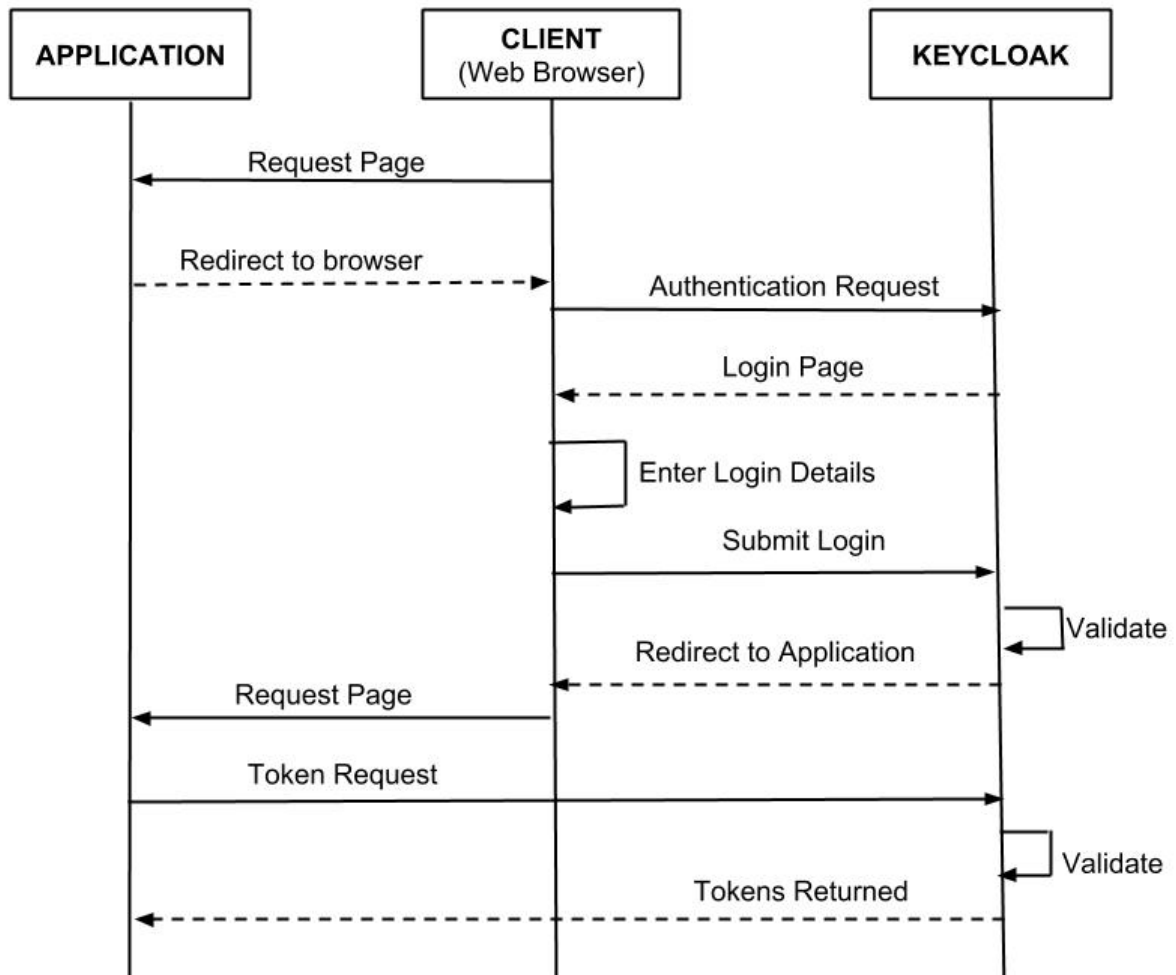


Keycloak With OpenID Connect (OIDC)

OIDC is an authentication protocol that is an extension of OAuth 2.0. OAuth 2.0 is only a framework for building authorisation protocols, but OIDC is a full-fledged authentication and authorisation protocol. OIDC authentication flow when integrated with keycloak:

- Browser visits application. The application notices the user is not logged in, so it redirects the browser to keycloak to be authenticated. The application passes along a call-back URL(a redirect URL) as a query parameter in this browser redirect that keycloak will use when it finishes authentication.
- Keycloak authenticates the user and creates a one-time, very short lived, temporary code. Keycloak redirects back to the application using the call-back URL provided earlier and additionally adds the temporary code as a query parameter in the call-back URL.

The application extracts the temporary code and makes a background out of band REST invocation to keycloak to exchange the code for an identity, access and refresh token. Once this temporary code has been used to obtain the tokens, it can never be used again. This prevents potential replay attacks.



Requirements

The tools that we require are mentioned below:

Operating systems: Windows, Docker

Tools: Maven

Single-Sign-On: Keycloak

Development environments: Eclipse

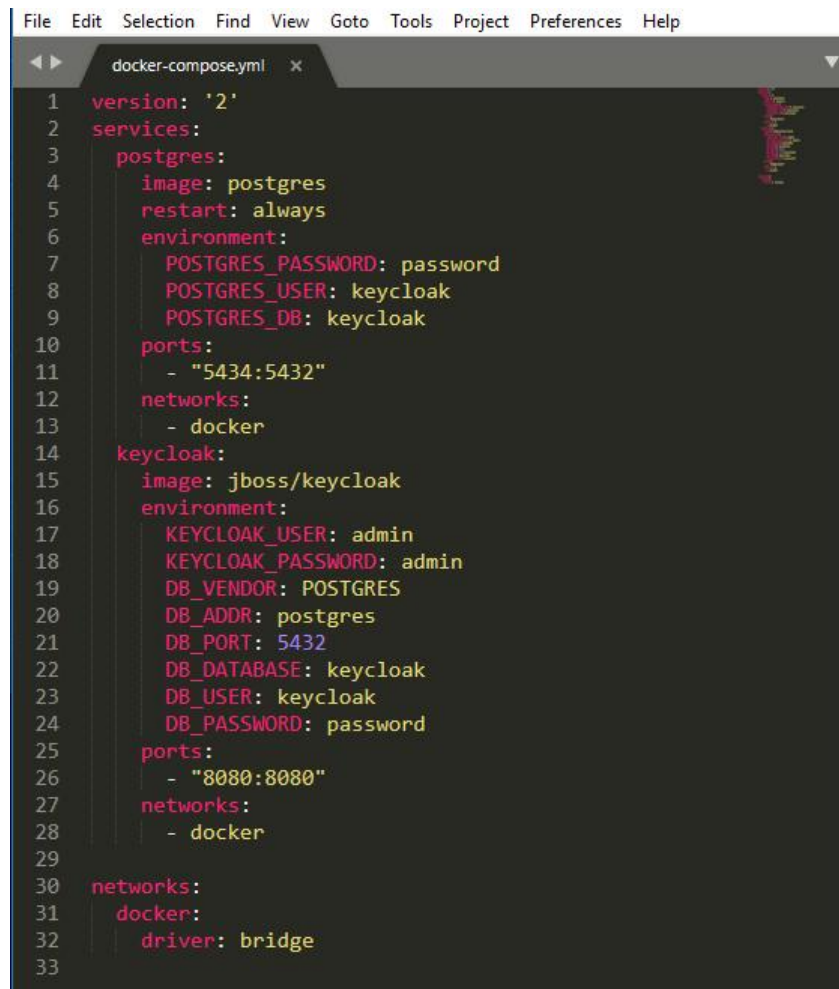
Programming Language : Java

Keycloak



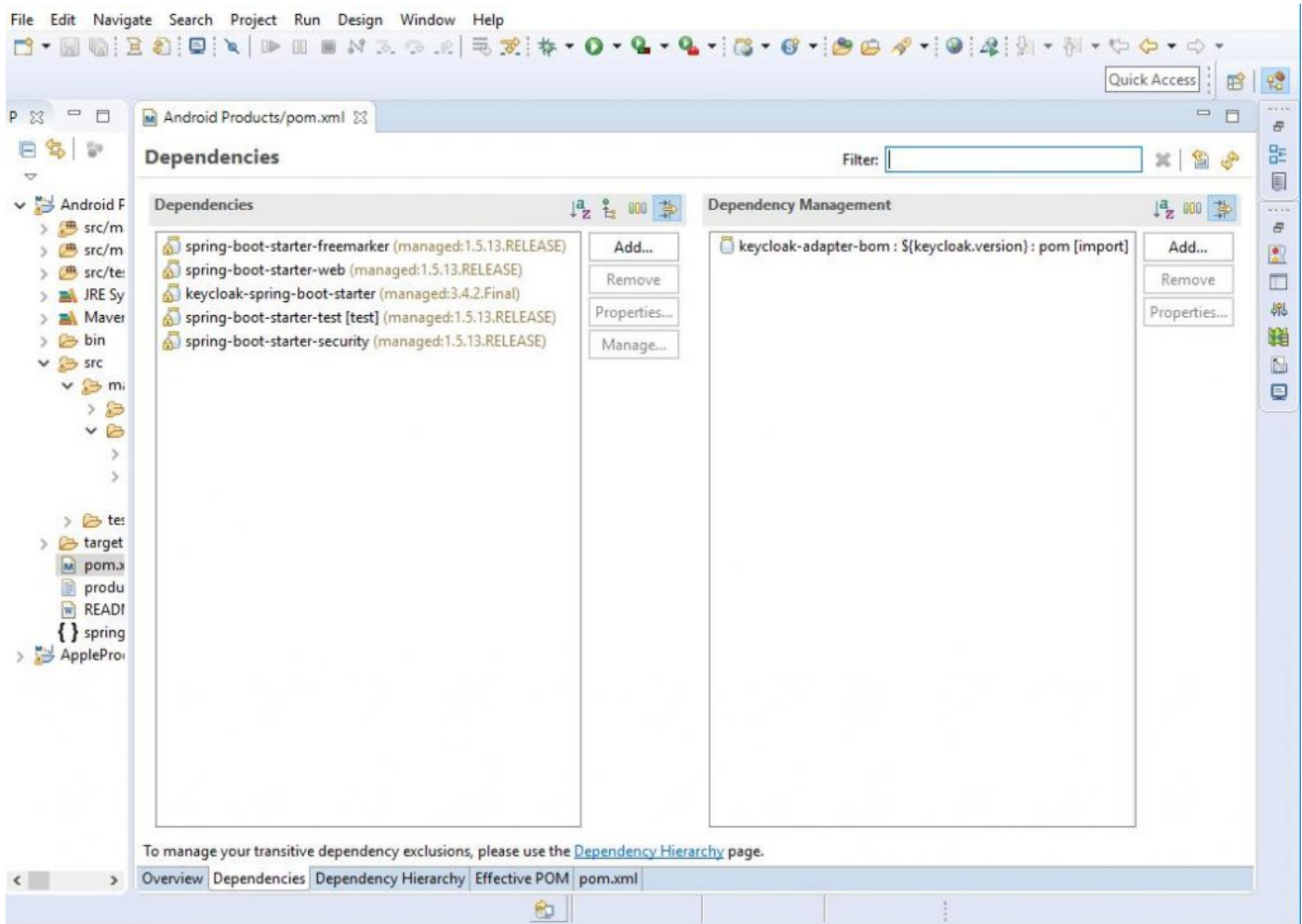
Integrating multiple applications with keycloak

1. Add docker-compose.yml file and save it in a folder.

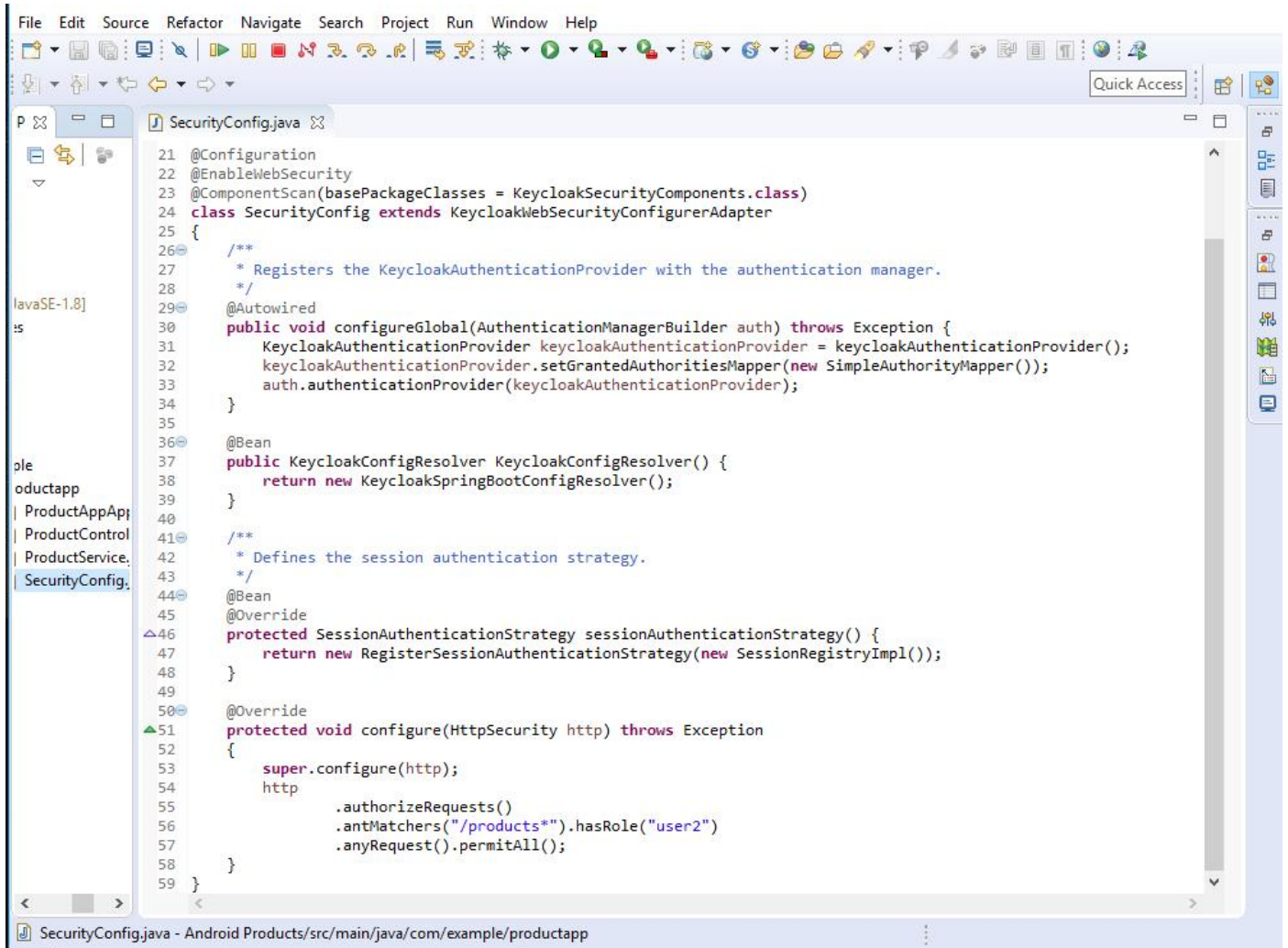


```
File Edit Selection Find View Goto Tools Project Preferences Help
docker-compose.yml x
1 version: '2'
2 services:
3   postgres:
4     image: postgres
5     restart: always
6     environment:
7       POSTGRES_PASSWORD: password
8       POSTGRES_USER: keycloak
9       POSTGRES_DB: keycloak
10    ports:
11      - "5434:5432"
12    networks:
13      - docker
14   keycloak:
15     image: jboss/keycloak
16     environment:
17       KEYCLOAK_USER: admin
18       KEYCLOAK_PASSWORD: admin
19       DB_VENDOR: POSTGRES
20       DB_ADDR: postgres
21       DB_PORT: 5432
22       DB_DATABASE: keycloak
23       DB_USER: keycloak
24       DB_PASSWORD: password
25     ports:
26       - "8080:8080"
27     networks:
28       - docker
29
30 networks:
31   docker:
32     driver: bridge
33
```

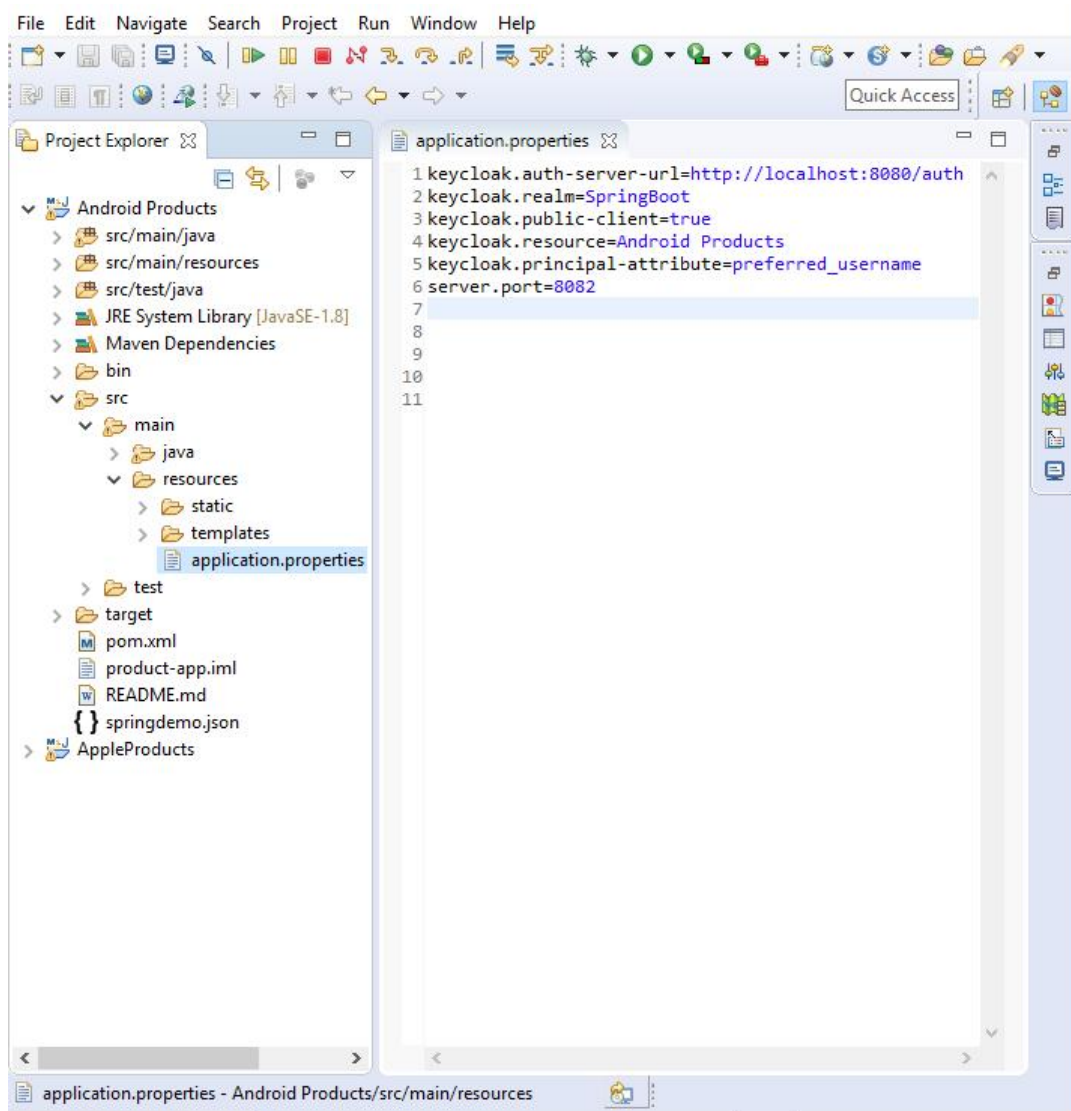
Add keycloak and spring-boot dependencies.



CreateSecurityConfig.java file to your project.



Create applications.properties file.

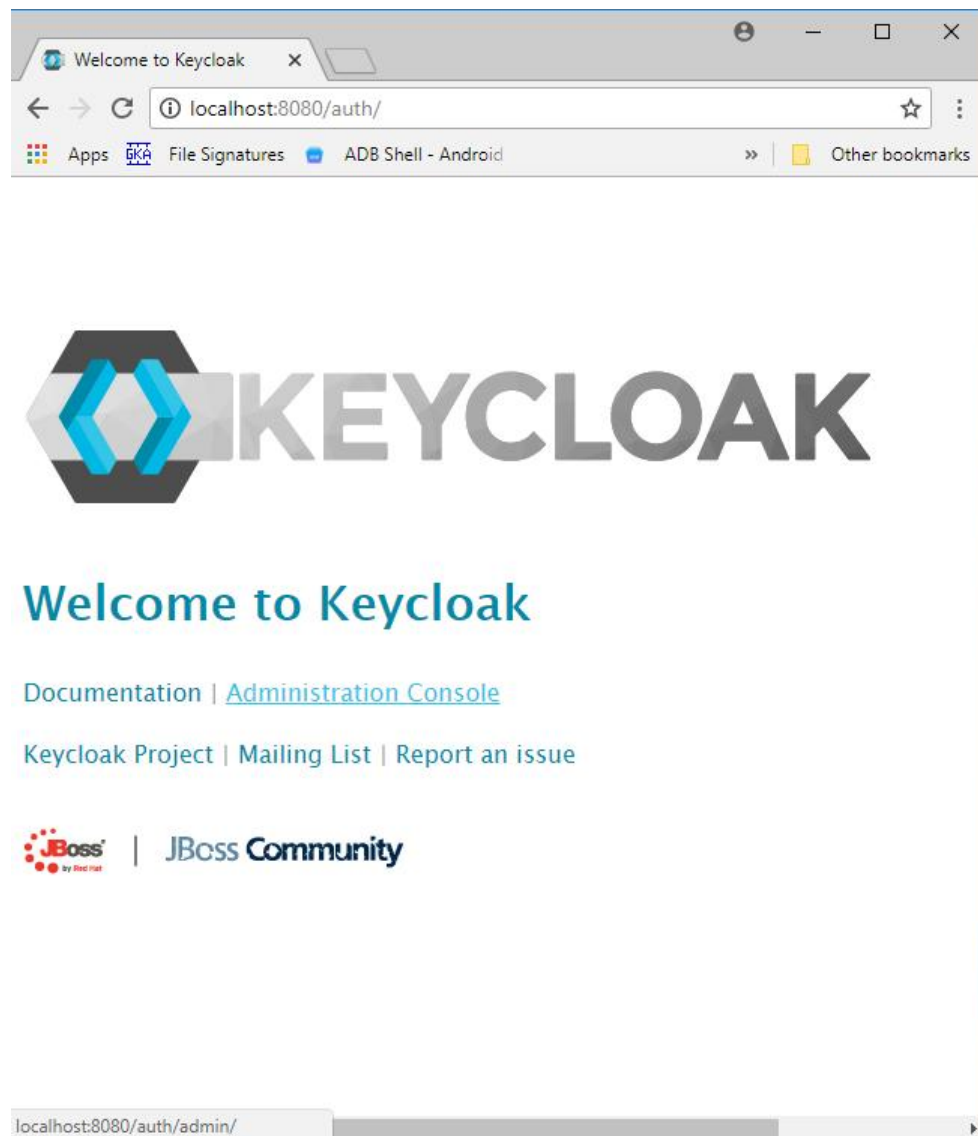


Open command prompt and run docker-compose up to run keycloak.

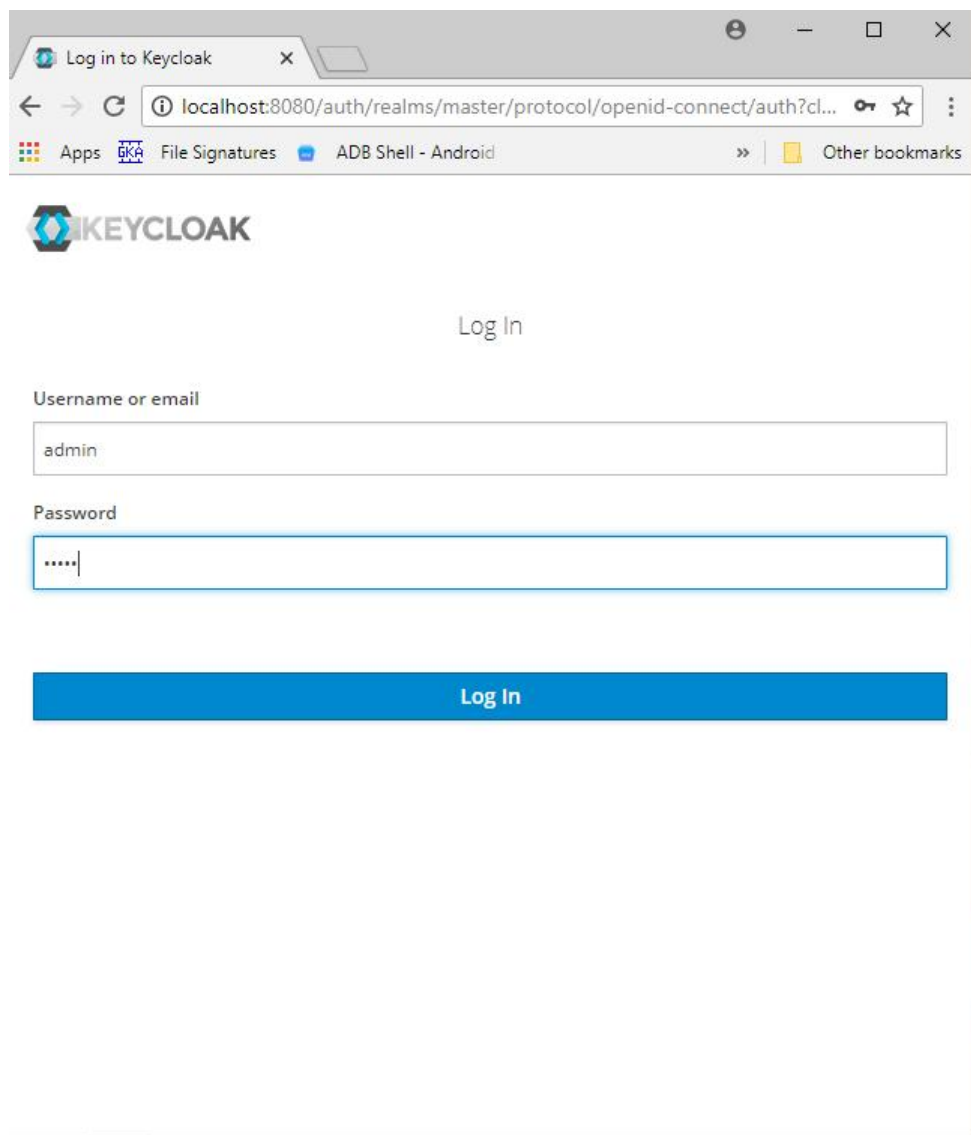
```
Command Prompt - docker-compose up

C:\Users\keycloak\Documents\develop>docker-compose up
Starting develop_postgres_1 ... done
Starting develop_keycloak_1 ... done
Attaching to develop_postgres_1, develop_keycloak_1
postgres_1 | 2018-07-17 11:23:44.781 UTC [1] LOG:  listening on IPv4 address "0.0.0.0",
postgres_1 | port 5432
postgres_1 | 2018-07-17 11:23:44.781 UTC [1] LOG:  listening on IPv6 address ":::", port
postgres_1 | 5432
postgres_1 | 2018-07-17 11:23:45.066 UTC [1] LOG:  listening on Unix socket "/var/run/po
postgres_1 | stgresql/.s.PGSQL.5432"
postgres_1 | 2018-07-17 11:23:45.307 UTC [20] LOG:  database system was shut down at 201
postgres_1 | 8-07-17 11:21:29 UTC
postgres_1 | 2018-07-17 11:23:45.370 UTC [1] LOG:  database system is ready to accept co
postgres_1 | nnections
keycloak_1 | User with username 'admin' already added to '/opt/jboss/keycloak/standalone
keycloak_1 | /configuration/keycloak-add-user.json'
keycloak_1 | =====
keycloak_1 | Using PostgreSQL database
keycloak_1 | =====
keycloak_1 |
```

Open browser and enter the keycloak url. Click on Administration Console.

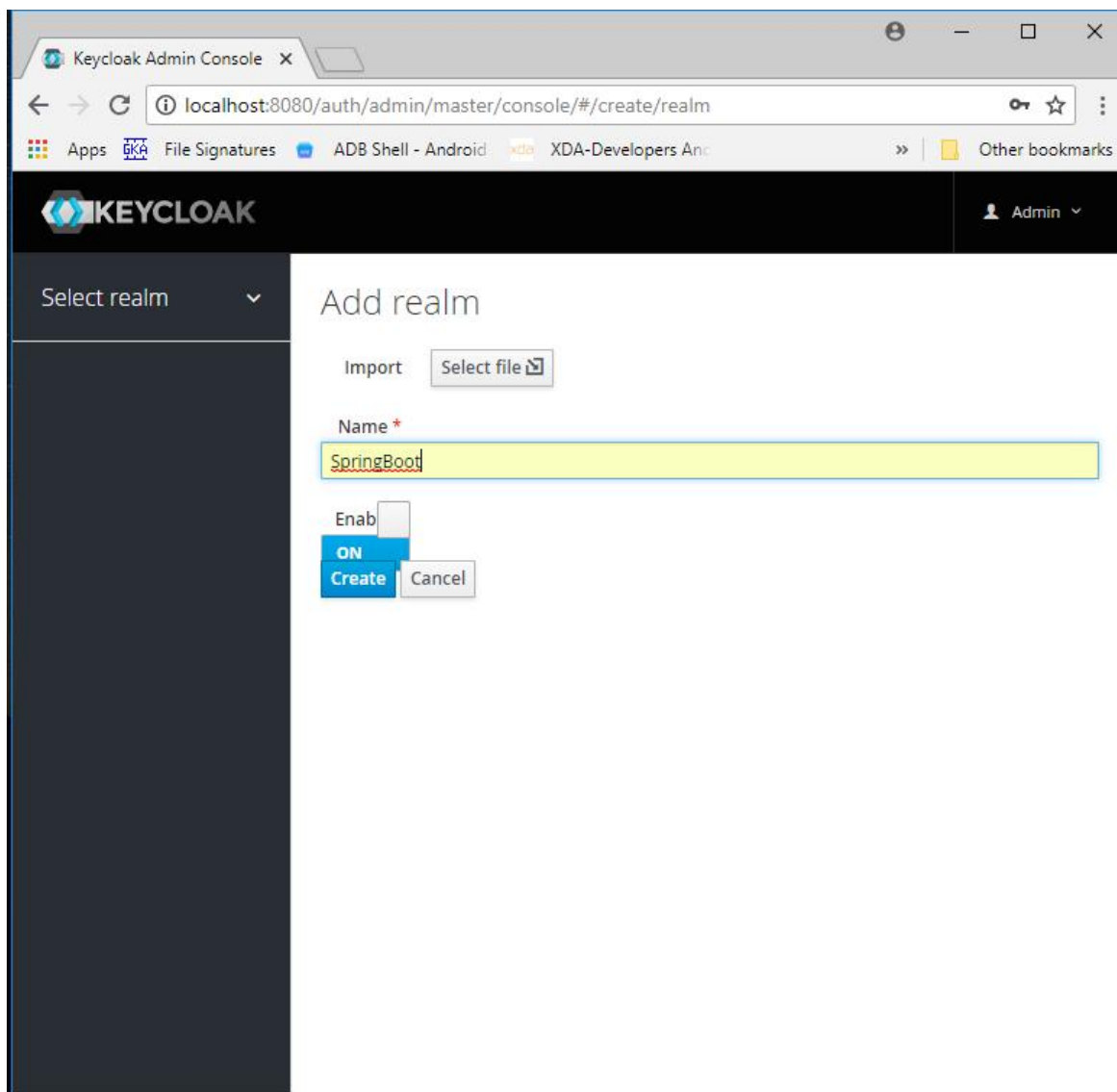


Default username and password is 'admin' , 'admin'.

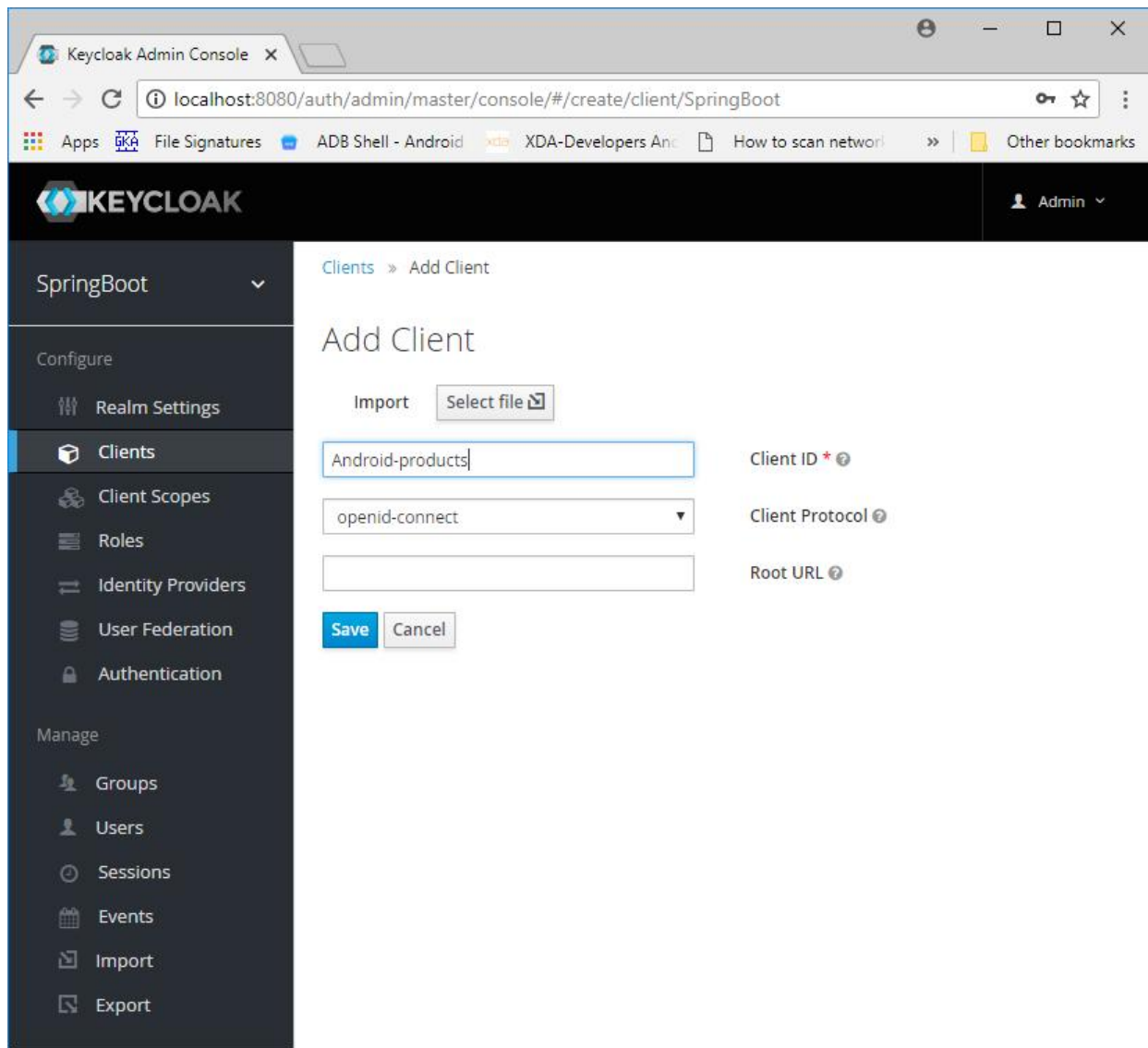


The screenshot shows a web browser window with the title "Log in to Keycloak". The address bar displays the URL "localhost:8080/auth/realms/master/protocol/openid-connect/auth?cl...". The browser's bookmark bar includes "Apps", "GKA", "File Signatures", "ADB Shell - Android", and "Other bookmarks". The Keycloak logo is at the top left of the page. The main heading is "Log In". Below it, there are two input fields: "Username or email" with the text "admin" and "Password" with masked characters ".....". A blue "Log In" button is positioned below the password field.

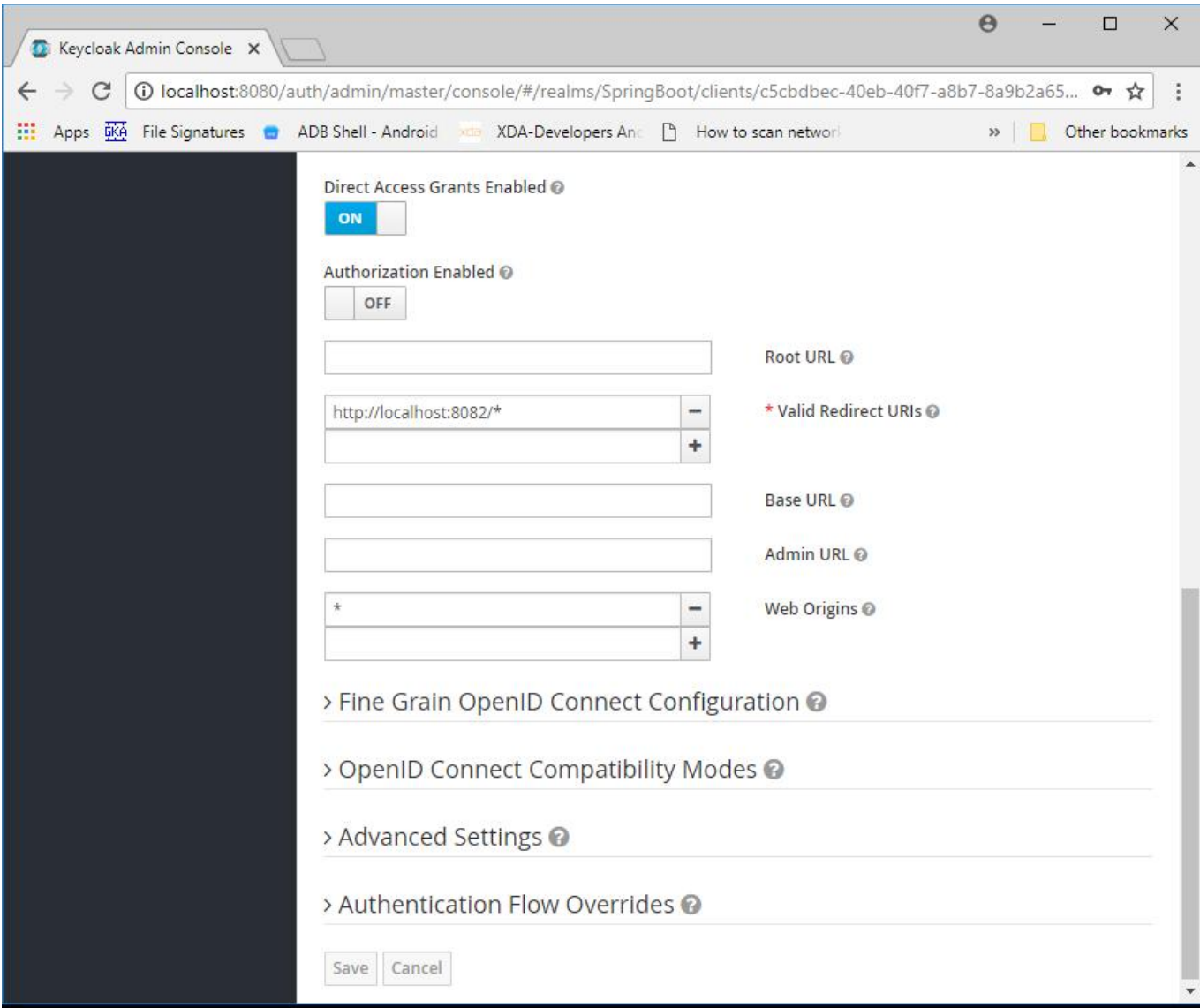
Click on Add Realm. Give a name to the realm.



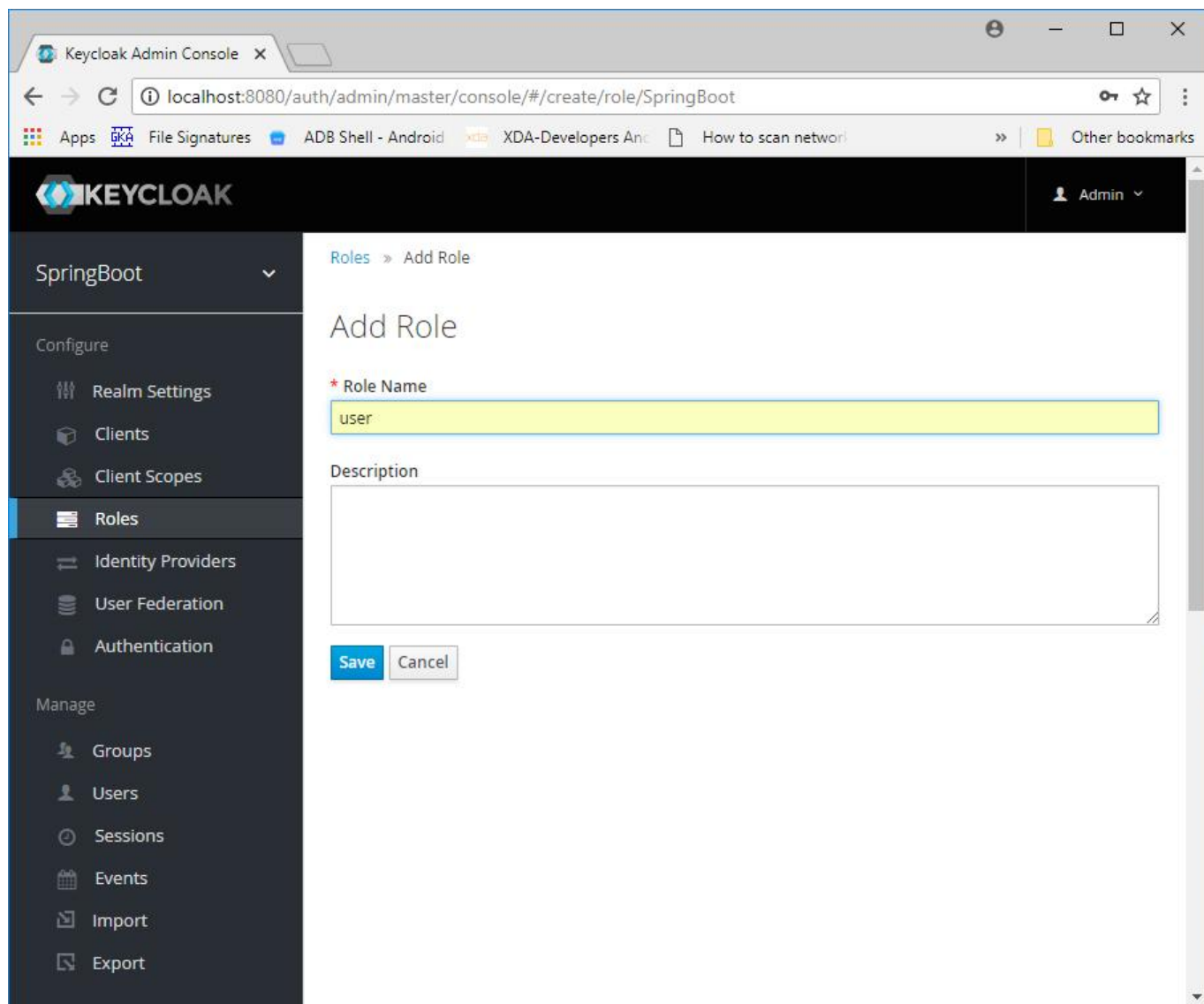
Add a client.



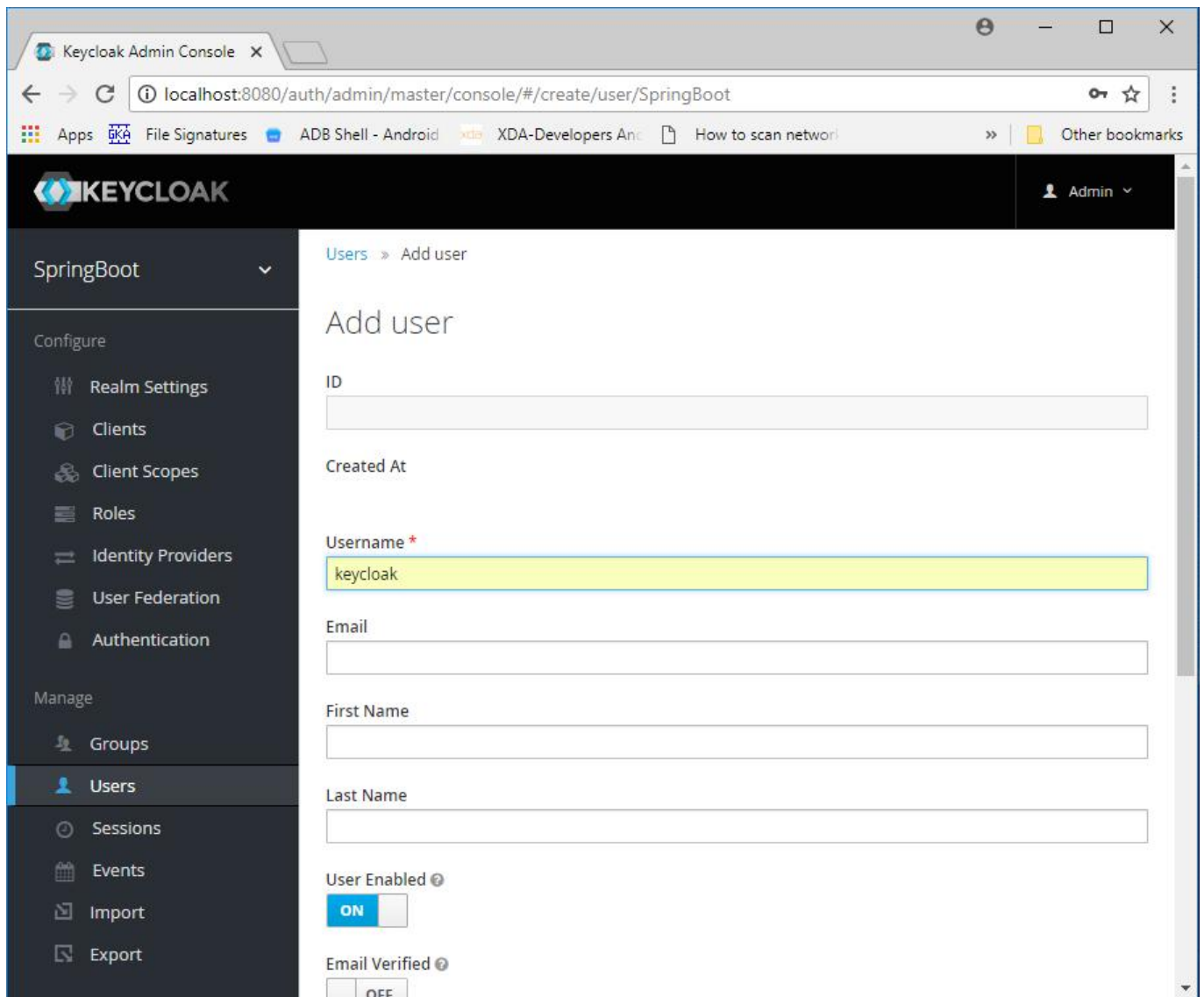
Give the URL path of your application in valid redirect URL .



Add role.



Add user.



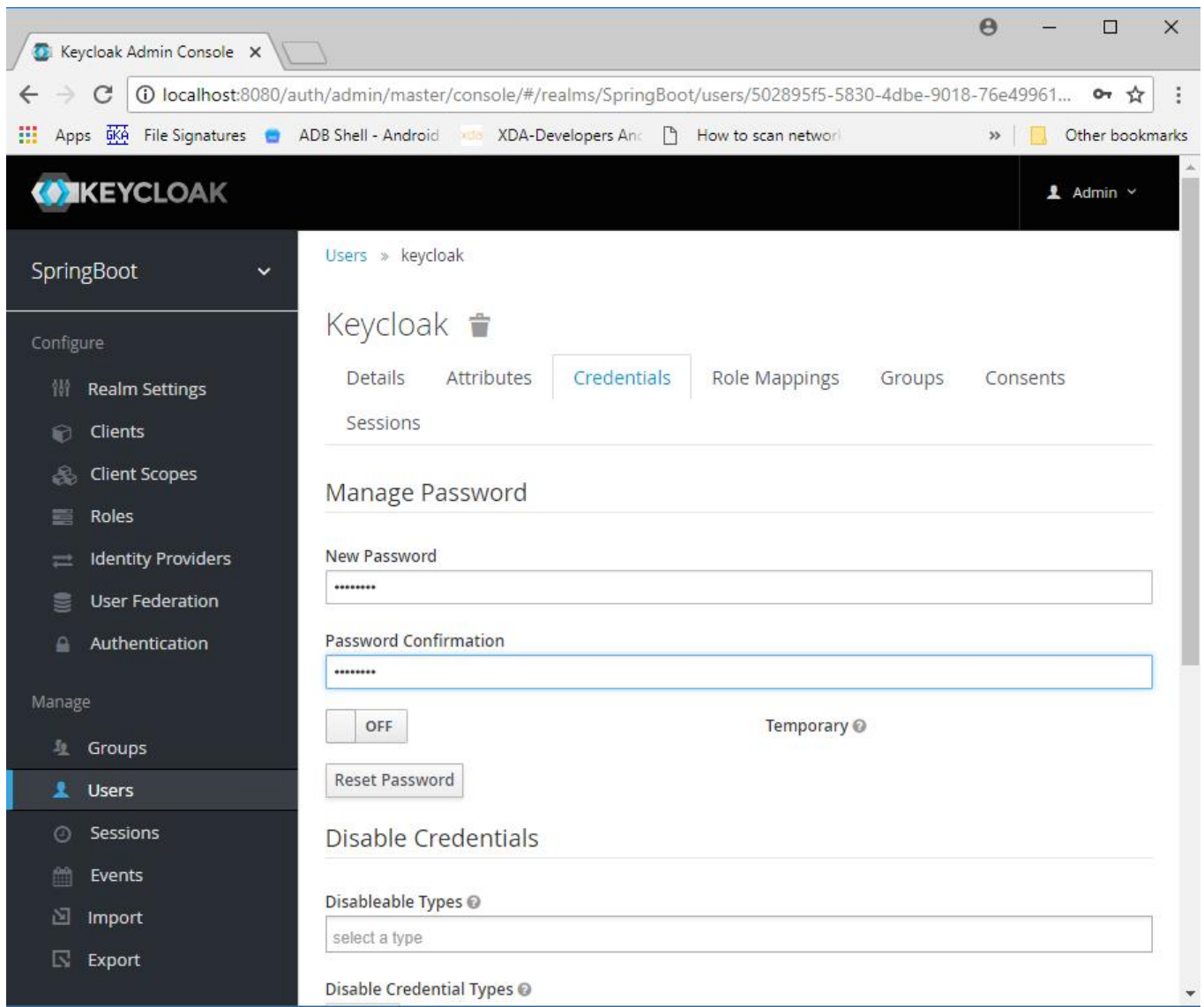
The screenshot shows the Keycloak Admin Console interface in a web browser. The browser's address bar displays the URL `localhost:8080/auth/admin/master/console/#/create/user/SpringBoot`. The page title is "Keycloak Admin Console". The left sidebar contains a navigation menu with the following sections:

- SpringBoot** (selected)
- Configure**
 - Realm Settings
 - Clients
 - Client Scopes
 - Roles
 - Identity Providers
 - User Federation
 - Authentication
- Manage**
 - Groups
 - Users** (selected)
 - Sessions
 - Events
 - Import
 - Export

The main content area is titled "Add user" and contains the following form fields:

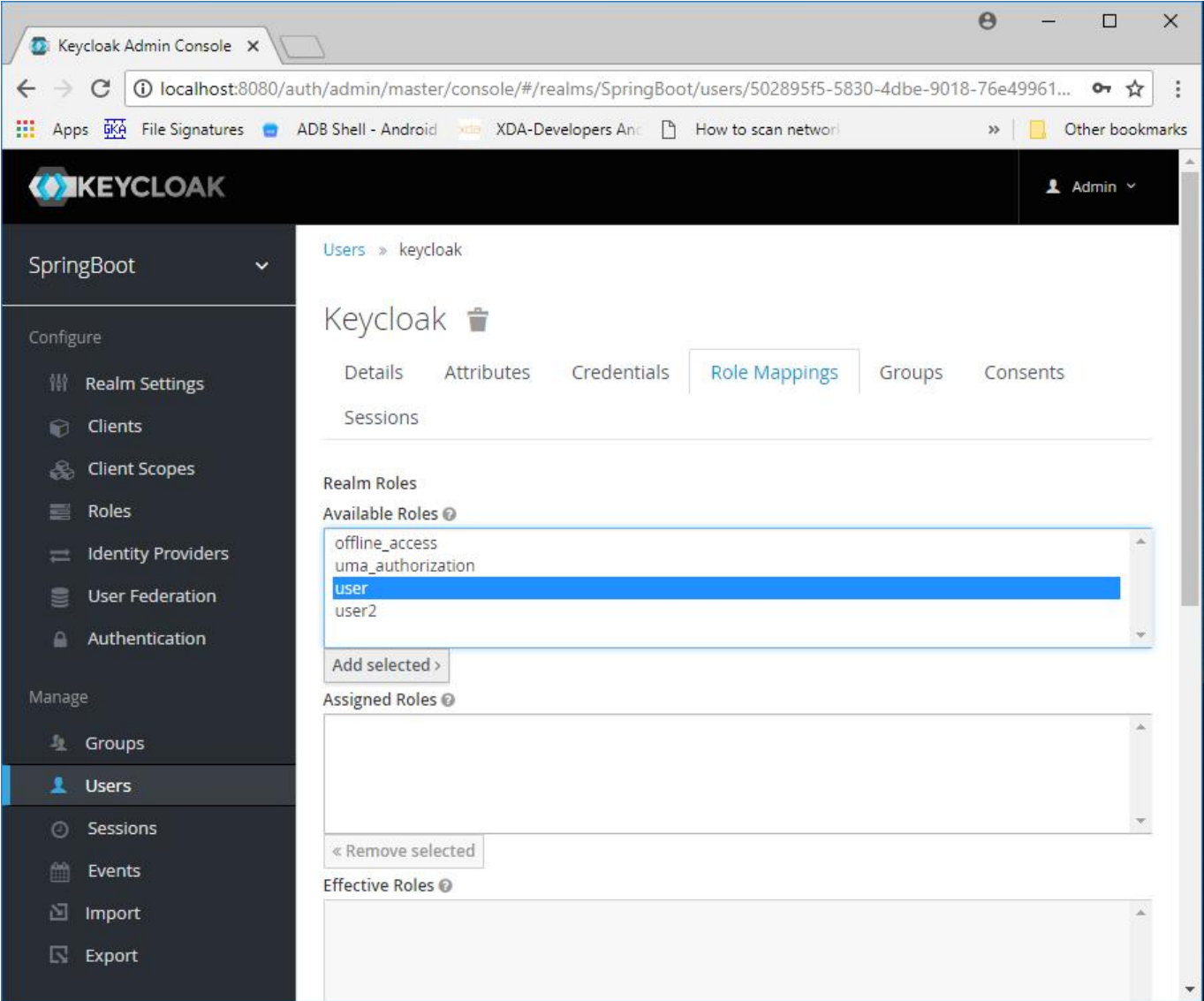
- ID:
- Created At:
- Username *:
- Email:
- First Name:
- Last Name:
- User Enabled: ☒ ON
- Email Verified: ☐ OFF

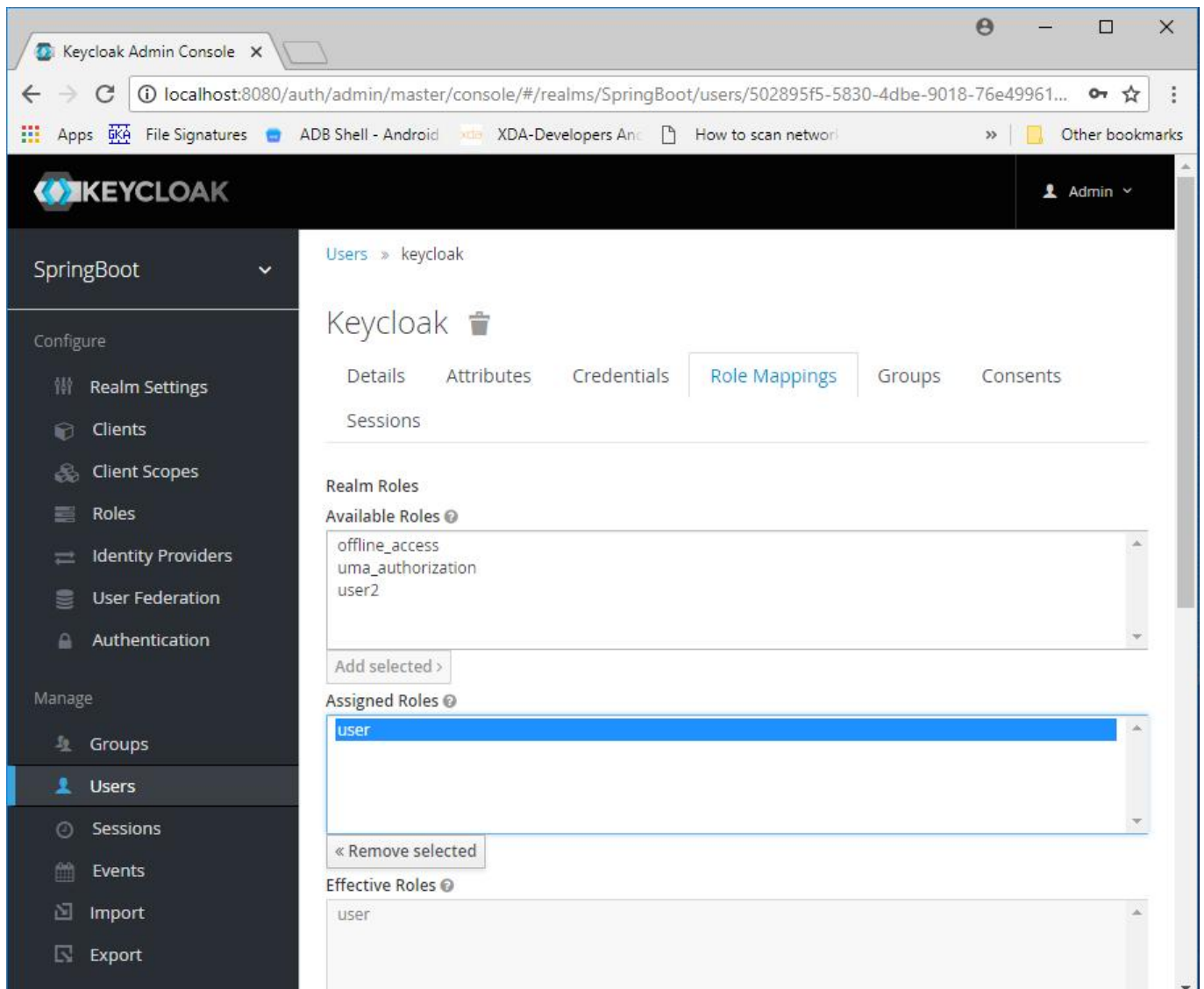
Go to credentials tab and reset password by giving new password.



The screenshot shows the Keycloak Admin Console interface. The browser address bar indicates the URL: `localhost:8080/auth/admin/master/console/#/realms/SpringBoot/users/502895f5-5830-4dbe-9018-76e49961...`. The left sidebar contains a navigation menu with sections: 'Configure' (Realm Settings, Clients, Client Scopes, Roles, Identity Providers, User Federation, Authentication) and 'Manage' (Groups, Users, Sessions, Events, Import, Export). The 'Users' section is selected. The main content area shows the 'Keycloak' user profile with tabs for 'Details', 'Attributes', 'Credentials' (active), 'Role Mappings', 'Groups', and 'Consents'. Below the tabs is a 'Sessions' section. The 'Manage Password' section includes a 'New Password' field, a 'Password Confirmation' field, a 'Temporary' toggle (currently OFF), and a 'Reset Password' button. Below this is a 'Disable Credentials' section with a 'Disableable Types' dropdown menu (showing 'select a type') and a 'Disable Credential Types' link.

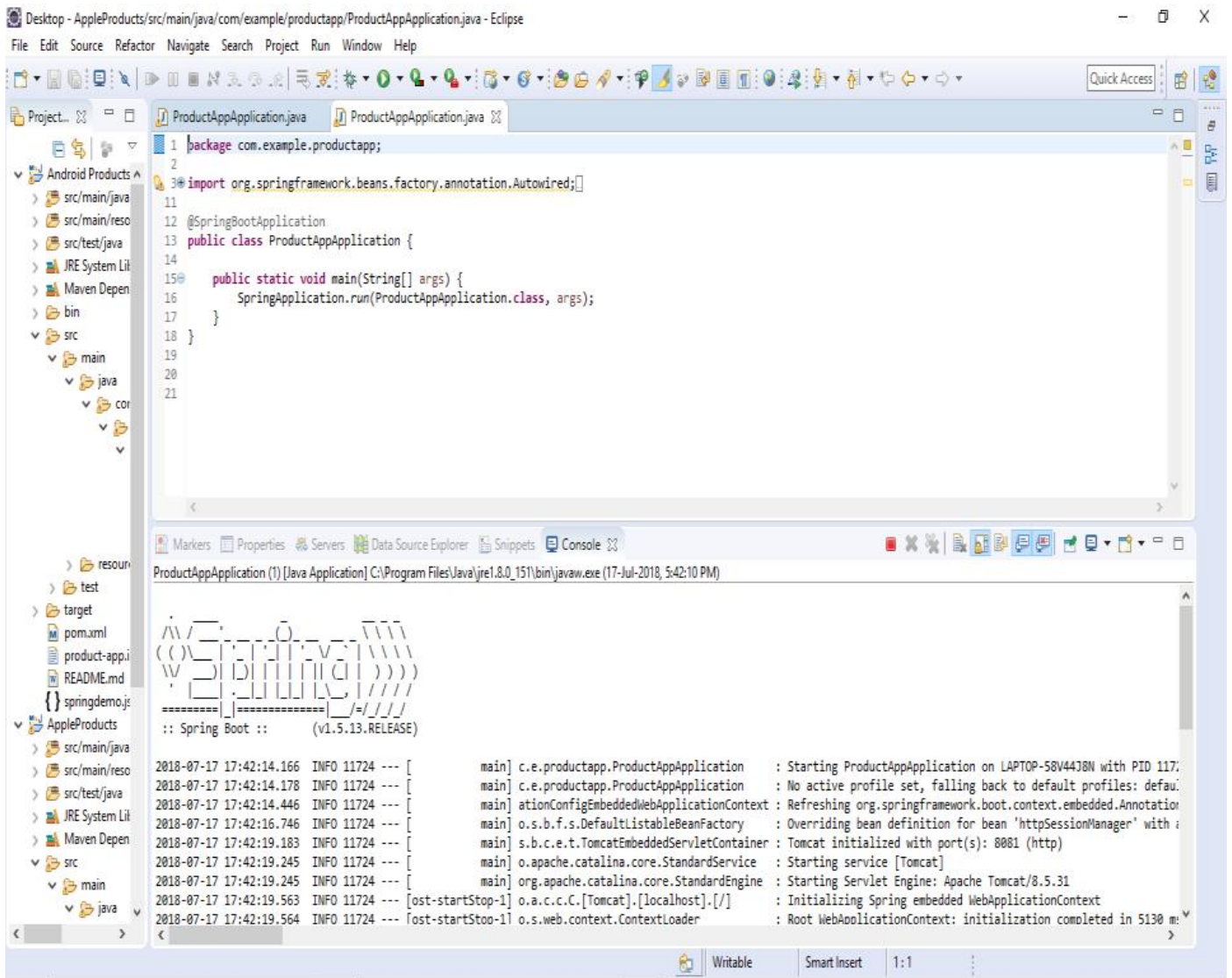
Go to role mappings tab and add a role to the user.



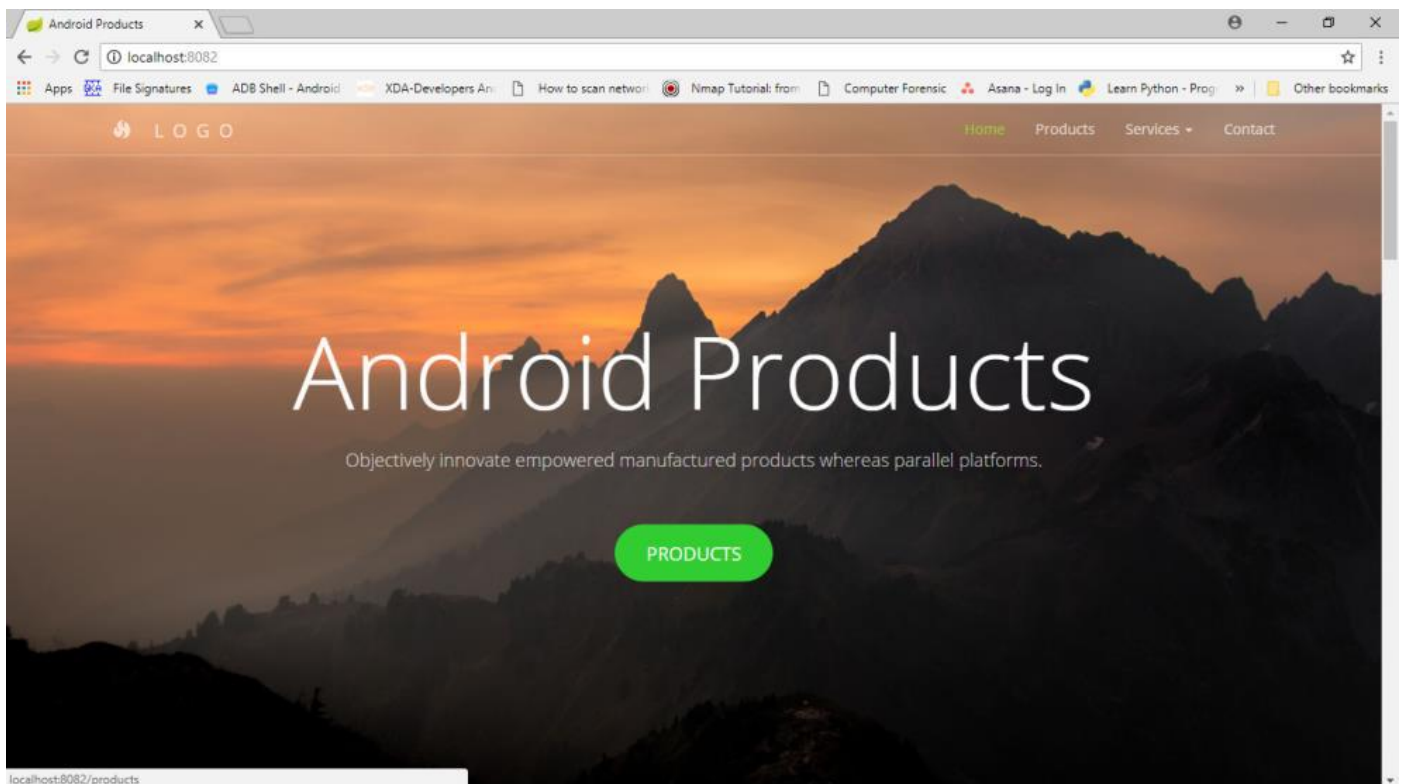


Similarly create another application and add all the necessary files. In keycloak we will use the same realm but we need to add new client, role and user.

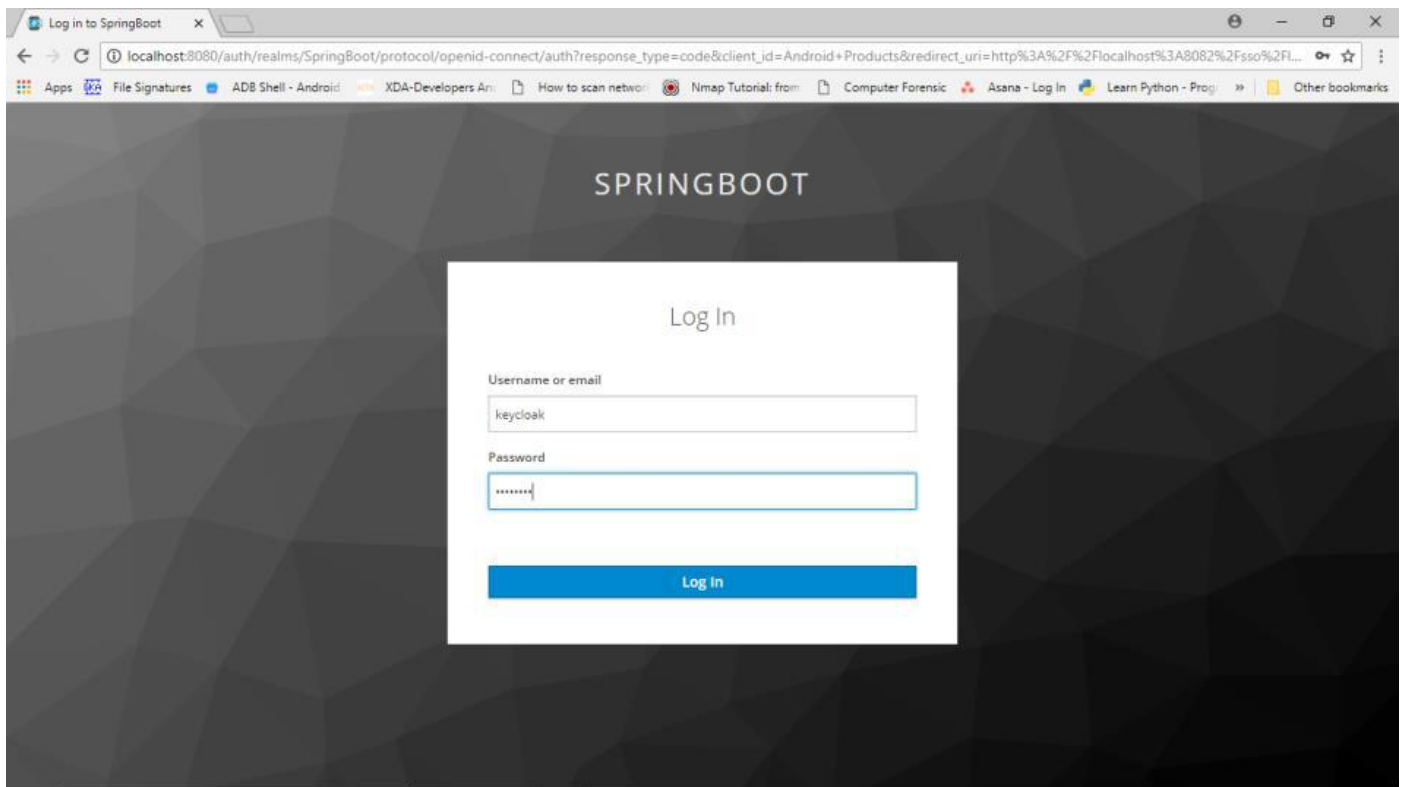
Now run two applications



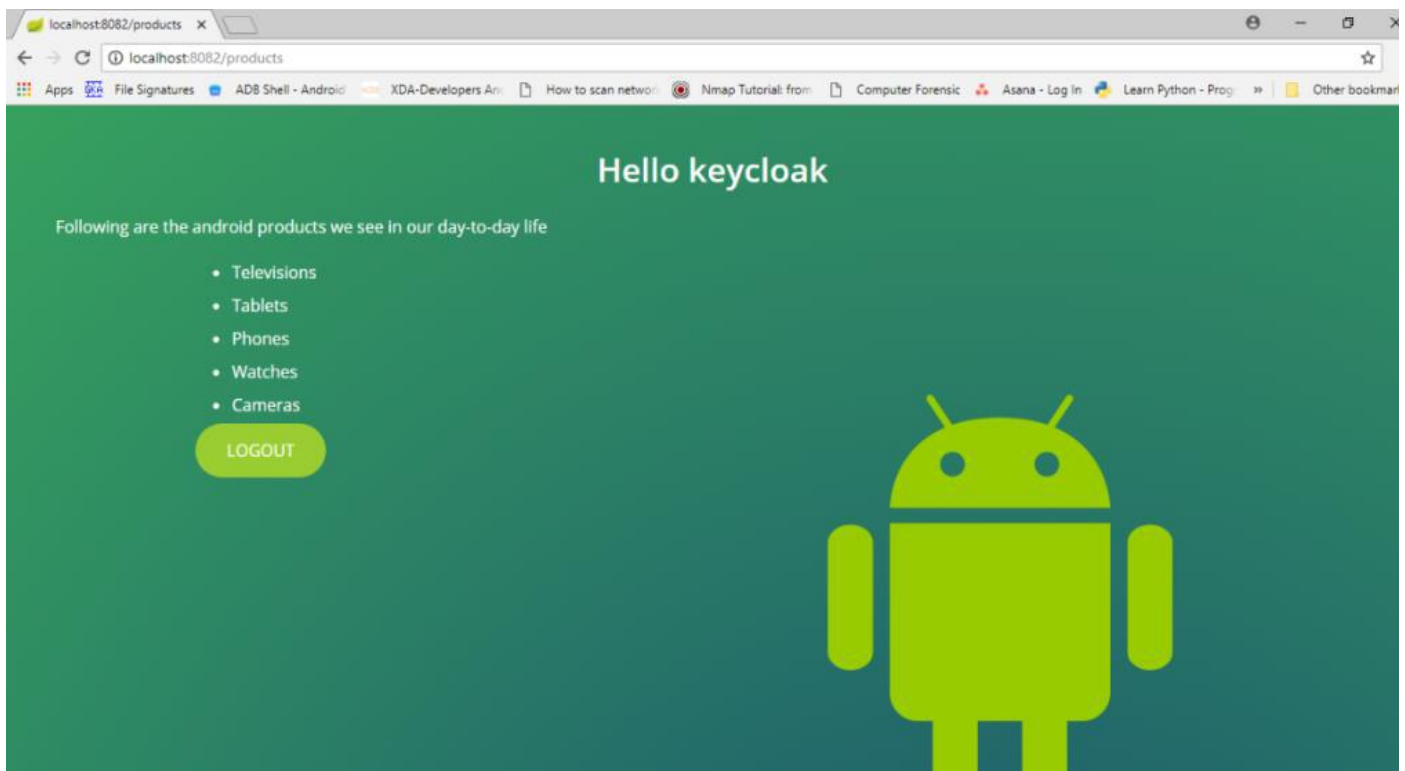
Open browser enter the url for first application.



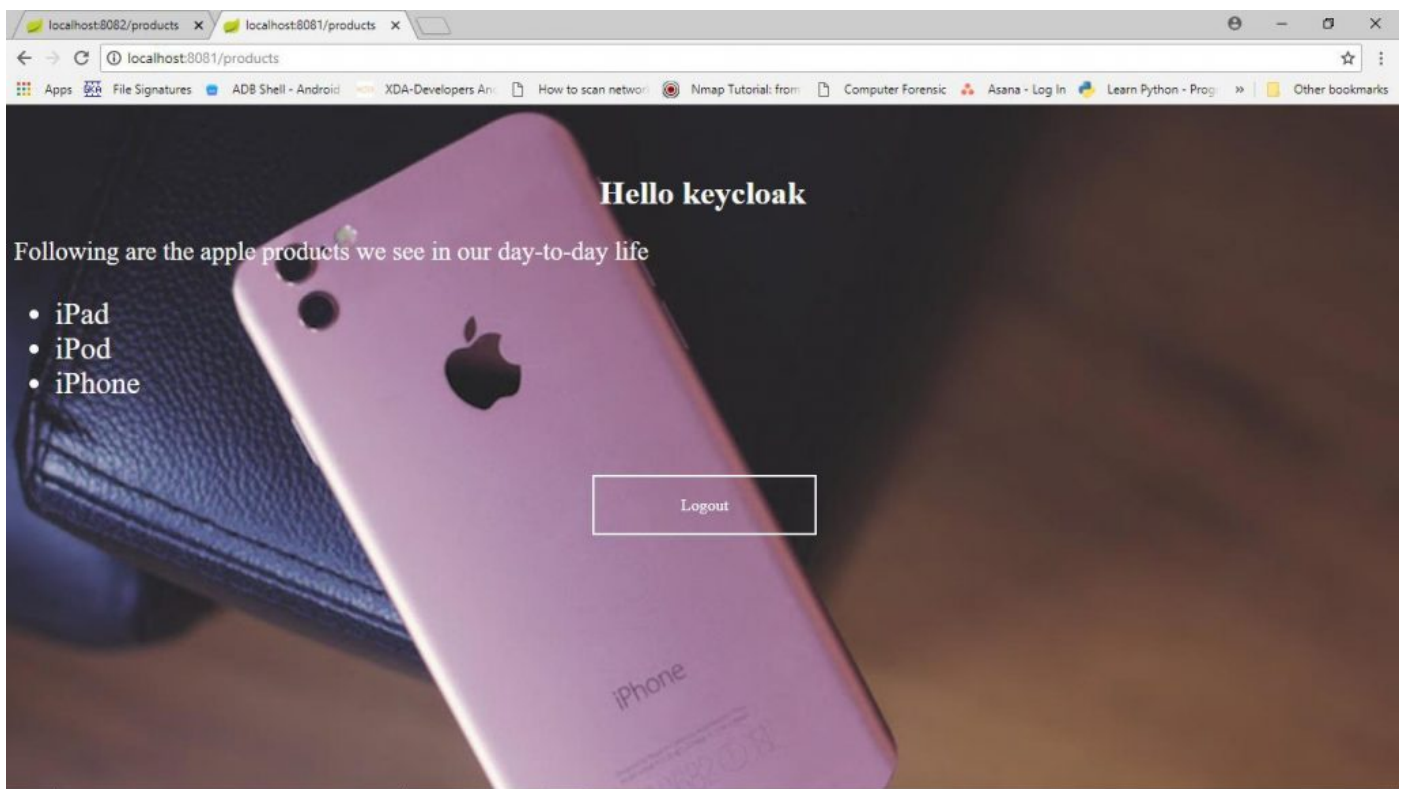
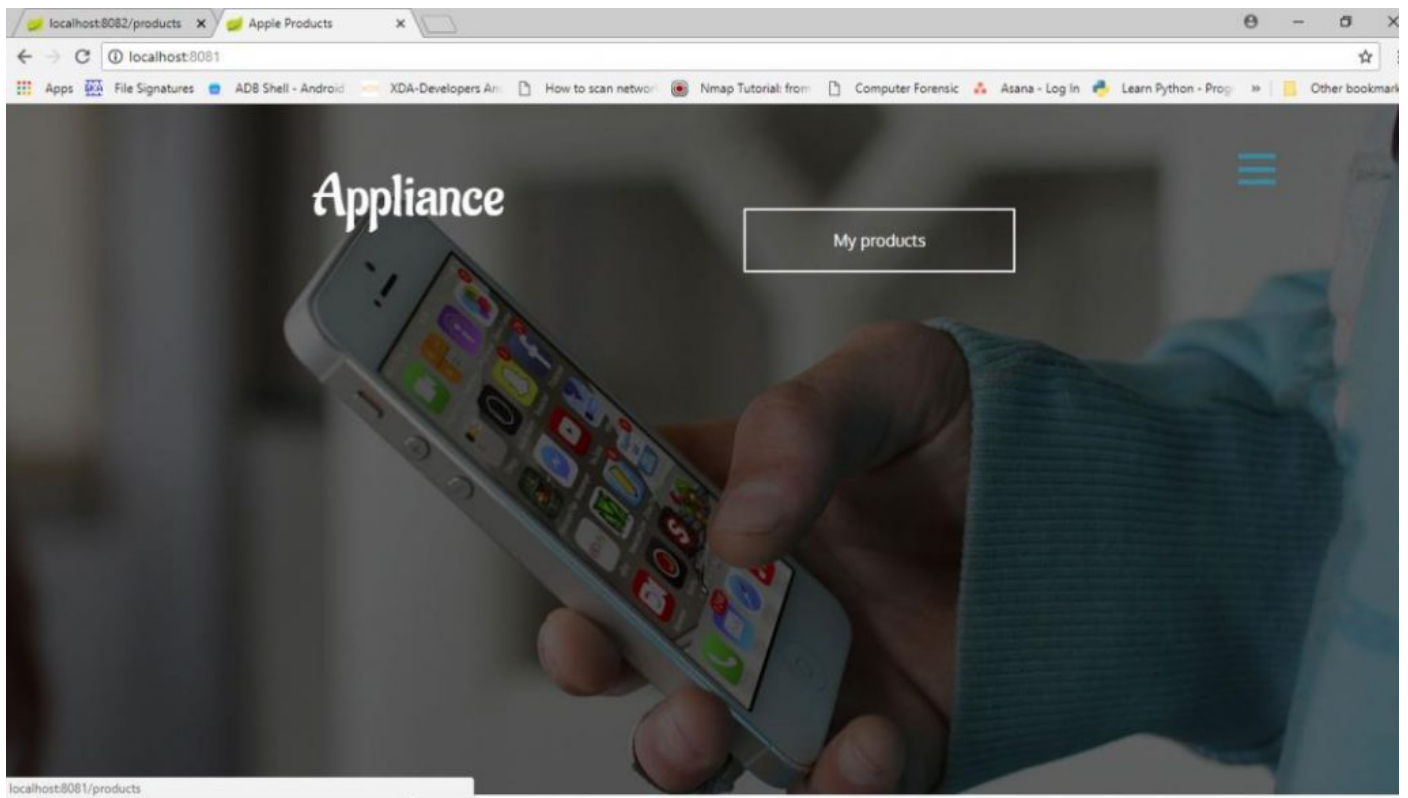
When I click on products button, it will redirect to keycloak. Give username and password.



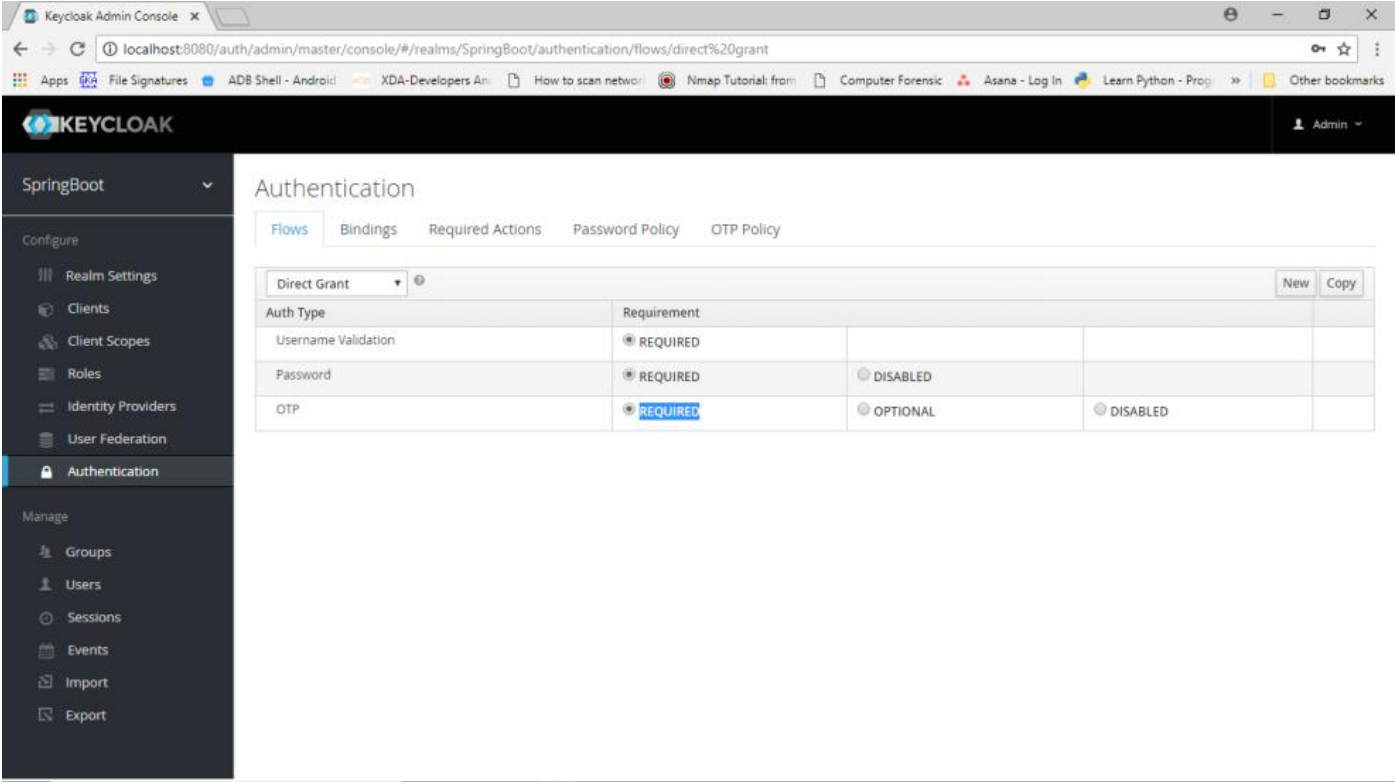
Now you will be redirected to the application.



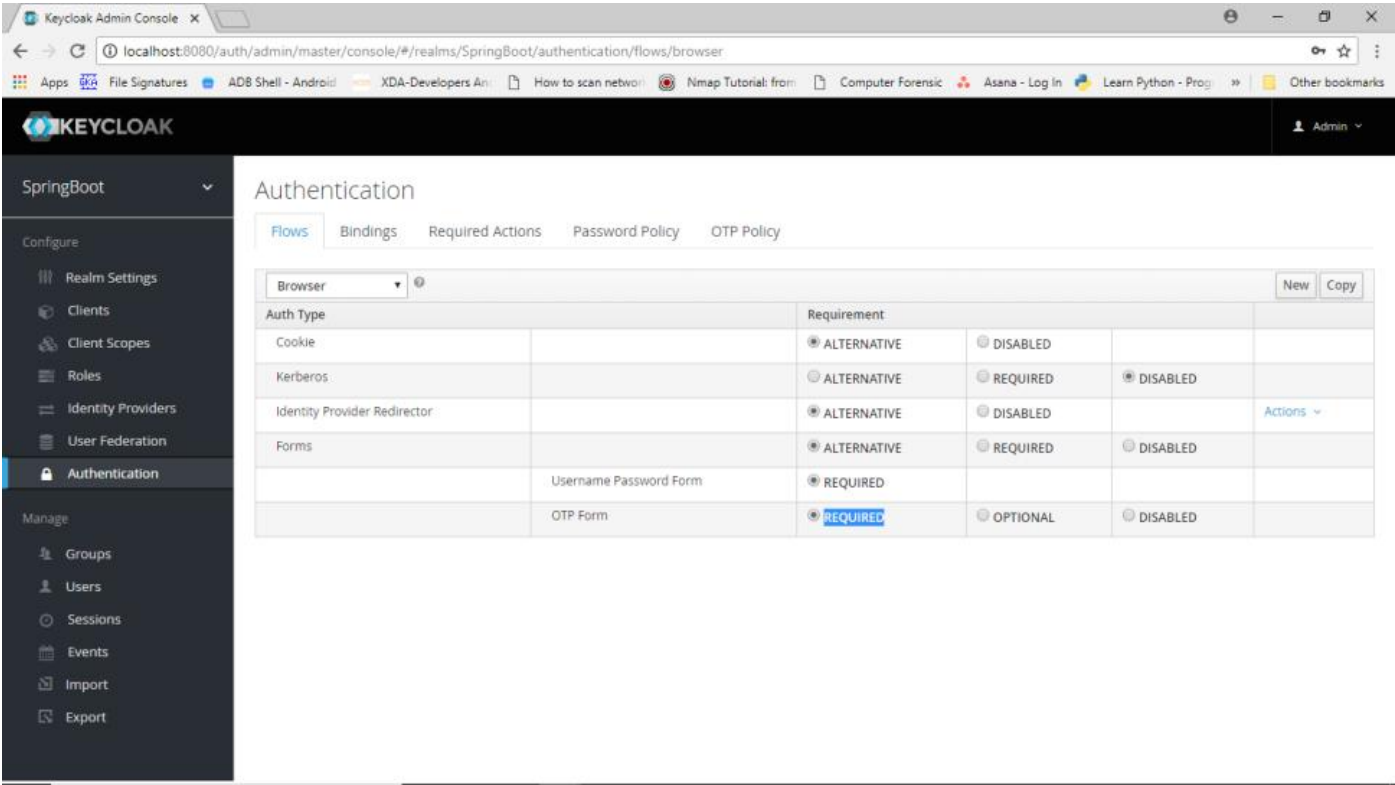
Now open another tab and give the URL for another application. You will be redirected to the application without going to the keycloak login screen.



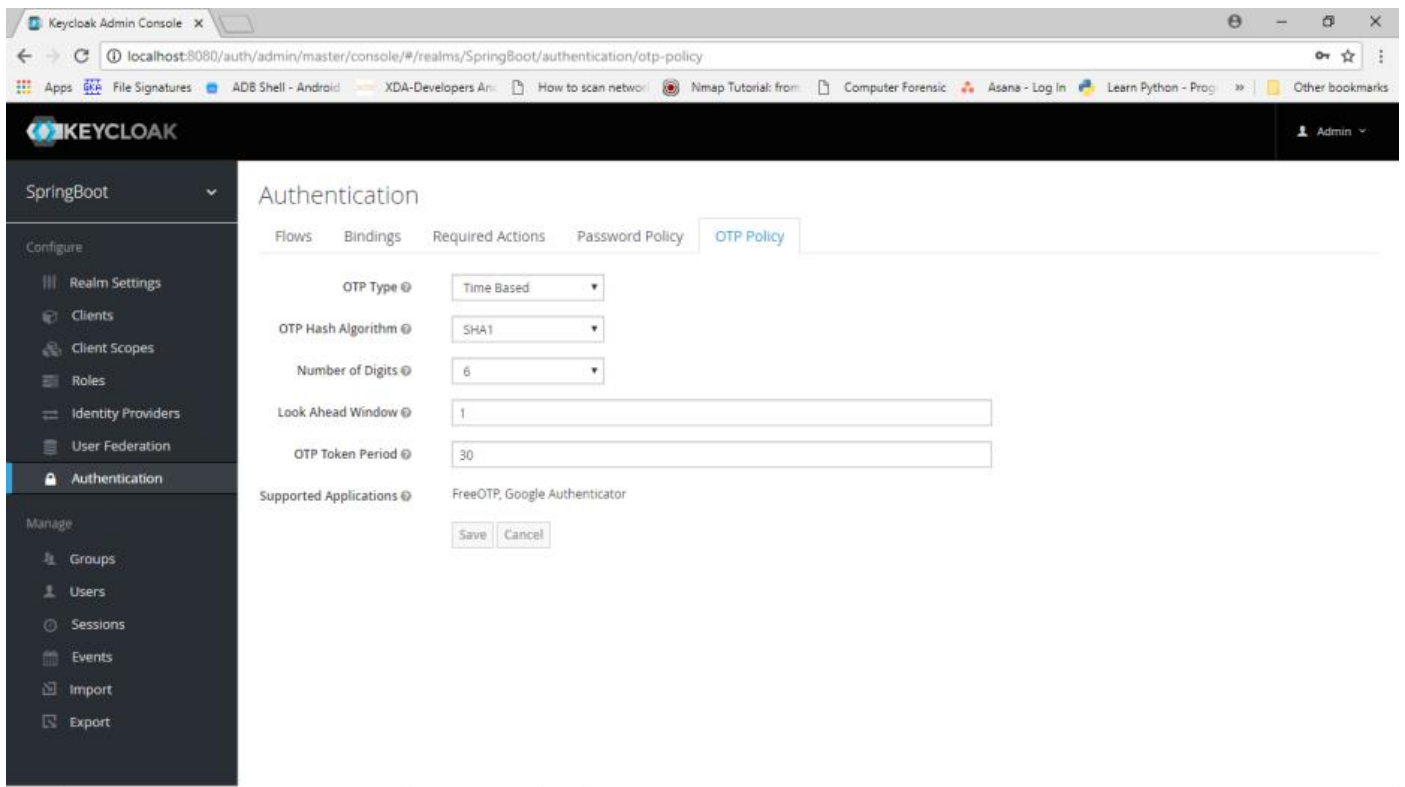
We will add an authentication method i.e. OTP. Go to authentication-flows-direct grant, at OTP mark it to required.



In browser flow mark OTP to required.

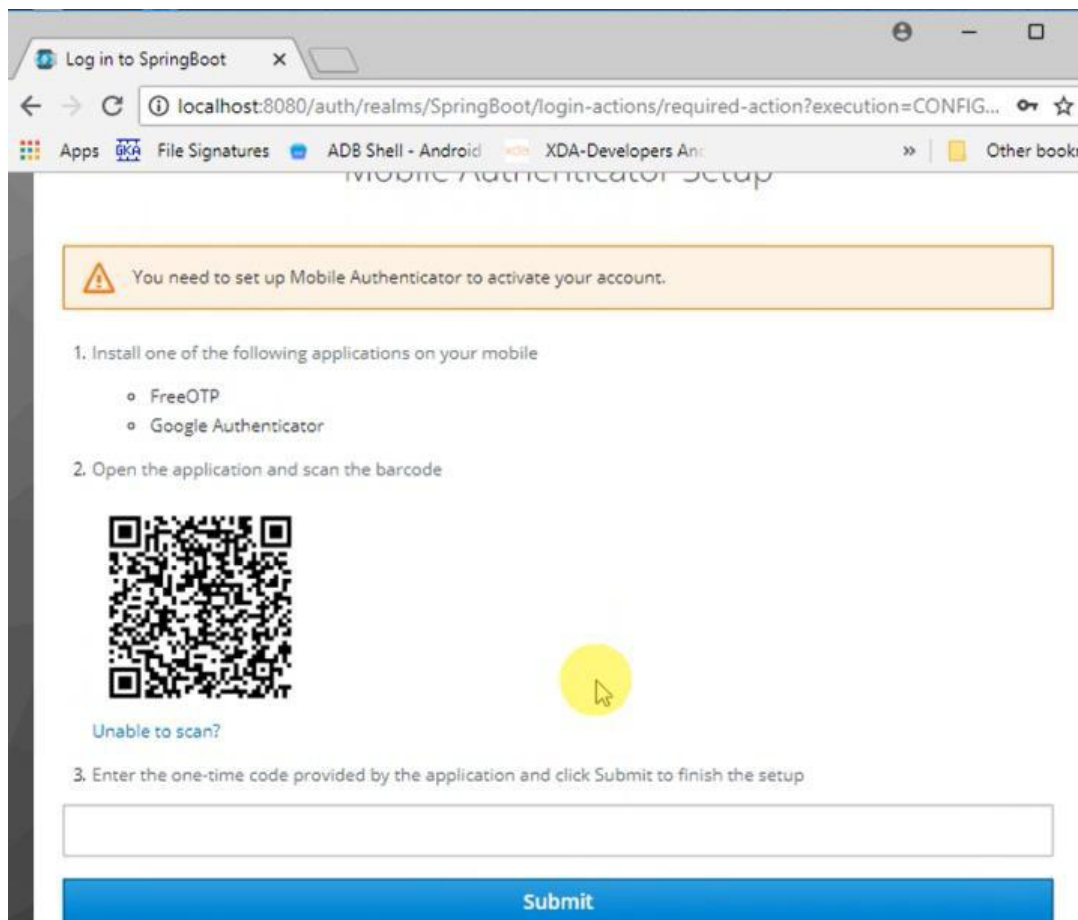


In OTP policy you can see the hash algorithms, time period etc.



You need to install FreeOTP application or Google Authenticator application on your mobile.

Now re-run keycloak, open browser and give a URL. After keycloak login page you will be redirected to this screen



Now scan the barcode with the application that you have installed on your mobile and enter the OTP.

You will be redirected to your application.

- GUIDE TO USING KEYCLOAK
- IDENTITY ACCESS MANAGEMENT
- KEYCLOAK
- KEYCLOAK ADMIN URL
- KEYCLOAK ARCHITECTURE
- KEYCLOAK AUTHENTICATION FLOW
- KEYCLOAK AUTHORIZATION
- KEYCLOAK AUTHORIZATION EXAMPLE
- KEYCLOAK DEMO
- KEYCLOAK FIRST LOGIN FLOW
- KEYCLOAK GUIDE
- KEYCLOAK LICENSE
- KEYCLOAK MEANING
- KEYCLOAK REFRESH TOKEN
- KEYCLOAK REVIEW
- KEYCLOAK SSO
- KEYCLOAK SSO TUTORIAL
- KEYCLOAK TUTORIAL
- KEYCLOAK VS LDAP
- REALM SCANS ACCESS DENIED
- WHAT IS KEYCLOAK
- WHAT IS KEYCLOAK USED FOR



COMAKEIT

coMakeIT, a software product engineering company. We accelerate product innovation, modernize aging applications, and productize best practices into new software IP.

ALL AUTHOR POSTS

Enter your keywords...

Q

CATEGORIES

Agile and Scrum	57
Application Modernization	10
Artificial Intelligence	3
Cloud Transformation	15
Collaborative Innovation	6
Dedicated Teams	7
DevOps	8
Digital Transformation	13
Distributed Innovation	7
IT industry	27
Legacy Modernization	13
Low code development	5
Mobile Development	21
Outsourcing & offshoring	42
Platform Engineering	3

Product Development	60
Product Engineering	17
Product Implementation	10
Product Innovation	19
Product Transformation	15
Software Development	46
Software QA	15
Techies Corner	20