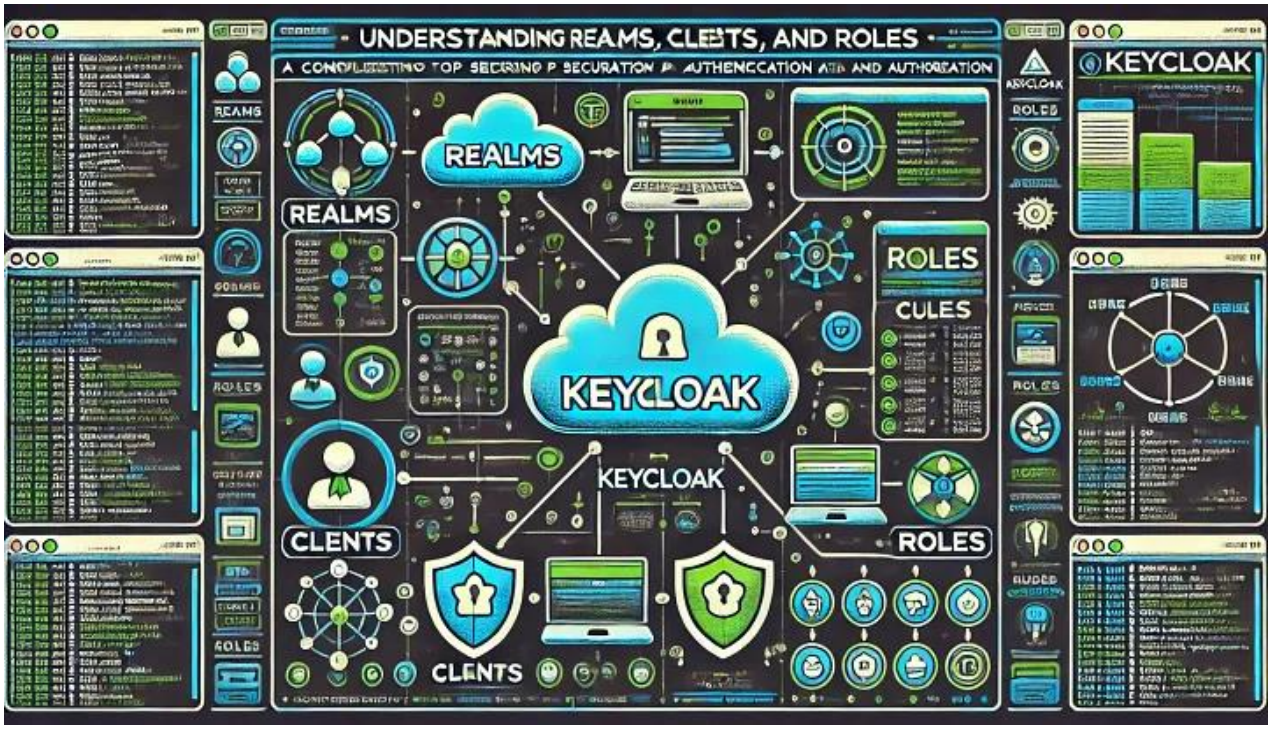
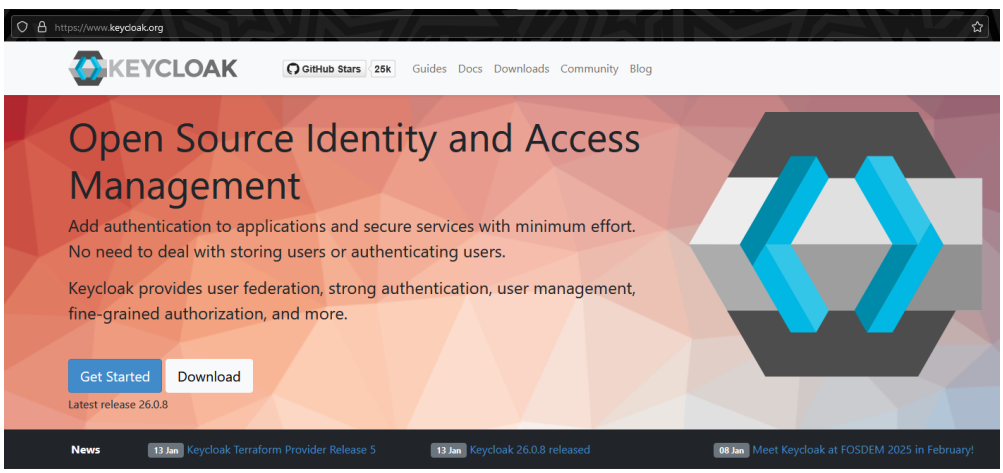


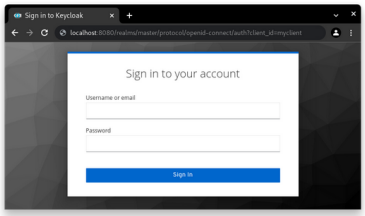
1



Single-Sign On

Users authenticate with Keycloak rather than individual applications. This means that your 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 a different application.

This also applies to logout. Keycloak provides single-sign out, which means users only have to logout once to be logged-out of all applications that use Keycloak.



2

Installation et démarrage de Keycloak

- Télécharger et extraire le ZIP de keycloak
  - https://www.keycloak.org/downloads
- Démarrer keycload avec la commande:kc.bat start-dev

KEYCLOAK GitHub Stars 25k Guides Docs Downloads Community Blog

Downloads 26.0.8

For a list of community maintained extensions check out the Extensions page.

Server

Keycloak	Distribution powered by Quarkus	ZIP (sha1) TAR.GZ (sha1)
Container image	For Docker, Podman, Kubernetes and OpenShift	Quay
Operator	For Kubernetes and OpenShift	OperatorHub
Third-party licenses	License and source code information for third-party dependencies	HTML

3

Administrator: Command Pro

```
C:\keycloak-26.0.8\bin>kc.bat start-dev --http-port=8180
2025-01-14 17:18:25,936 INFO  [org.keycloak.quarkus.runtime.storage.infinispan.CacheManagerFactory] (main) Starting Infi
nispan embedded cache manager
2025-01-14 17:18:26,245 INFO  [org.keycloak.quarkus.runtime.storage.infinispan.CacheManagerFactory] (main) Persistent us
er sessions enabled and no memory limit found in configuration. Setting max entries for sessions to 10000 entries.
2025-01-14 17:18:26,245 INFO  [org.keycloak.quarkus.runtime.storage.infinispan.CacheManagerFactory] (main) Persistent us
er sessions enabled and no memory limit found in configuration. Setting max entries for clientSessions to 10000 entries.
2025-01-14 17:18:26,245 INFO  [org.keycloak.quarkus.runtime.storage.infinispan.CacheManagerFactory] (main) Persistent us
er sessions enabled and no memory limit found in configuration. Setting max entries for offlineSessions to 10000 entries
.
2025-01-14 17:18:26,246 INFO  [org.keycloak.quarkus.runtime.storage.infinispan.CacheManagerFactory] (main) Persistent us
er sessions enabled and no memory limit found in configuration. Setting max entries for offlineClientSessions to 10000 e
ntries.
2025-01-14 17:18:26,784 INFO  [org.infinispan.CONTAINER] (ForkJoinPool.commonPool-worker-1) ISPN000556: Starting user ma
rshaller 'org.infinispan.commons.marshall.ImmutableProtoStreamMarshaller'
2025-01-14 17:18:27,682 INFO  [org.keycloak.broker.provider.AbstractIdentityProviderMapper] (main) Registering class org
.keycloak.broker.provider.mappersync.ConfigSyncEventListener
2025-01-14 17:18:27,848 INFO  [org.keycloak.connections.infinispan.DefaultInfinispanConnectionProviderFactory] (main) No
de name: node_738345, Site name: null
2025-01-14 17:18:29,005 WARN  [io.agroal.pool] (main) Datasource '<default>': JDBC resources leaked: 1 ResultSet(s) and
0 Statement(s)
2025-01-14 17:18:29,241 INFO  [io.quarkus] (main) Keycloak 26.0.8 on JVM (powered by Quarkus 3.15.1) started in 7.421s.
Listening on: http://0.0.0.0:8180
2025-01-14 17:18:29,278 INFO  [io.quarkus] (main) Profile dev activated.
2025-01-14 17:18:29,279 INFO  [io.quarkus] (main) Installed features: [agroal, cdi, hibernate-orm, jdbc-h2, keycloak, na
rayana-jta, opentelemetry, reactive-routes, rest, rest-jackson, smallrye-context-propagation, vertx]
2025-01-14 17:18:29,286 WARN  [org.keycloak.quarkus.runtime.KeycloakMain] (main) Running the server in development mode.
 DO NOT use this configuration in production.
2025-01-14 17:19:51,949 INFO  [org.keycloak.services] (executor-thread-1) KC-SERVICES0077: Created temporary admin user
with username xproce
```

4

Installation et démarrage de Keycloak

- Par défaut, keycloak dispose d'une interface web d'administration accessible sur le port 8080.
  - kc.bat start-dev --http-port=8180
- La console d'administration permet d'administrer keycloak
- À la première utilisation de la console d'administration, il faut créer un utilisateur administrateur de du serveur keycloak

5

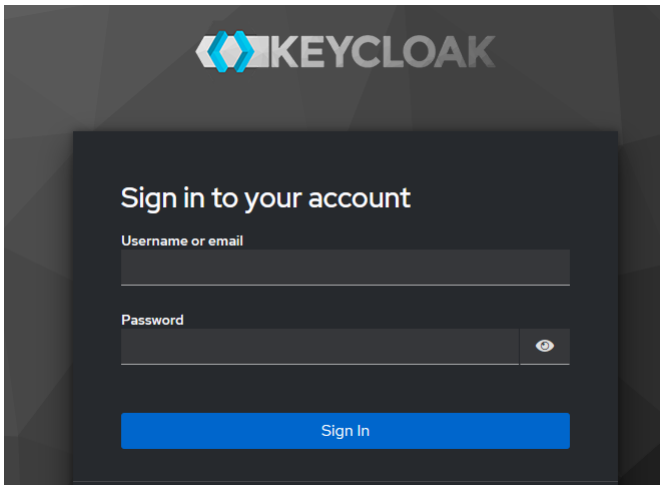
Architecture de keycloak

https://www.keycloak.org/docs/latest/server_admin/

- Realms (Royaumes): un domaine qui gère un ensemble d'utilisateurs, d'informations d'identification, des rôles et des groupes.
- Users : Ils sont des entités pouvant se connecter à votre système
- Rôles : ils identifient un type ou une catégorie d'utilisateur. Admin, user, …
- Groups : permet de gérer un groupe d'utilisateur
- Clients : ils sont des entités pouvant demander à Keycloak d'authentifier un utilisateur (comme des applications)
- Identity token: un token qui fournit des informations d'identité sur l'utilisateur
- Access token : un token pouvant être fourni dans le cadre d'une requête HTTP autorisant l'accès au service invoqué.
- Thèmes : Les thèmes et styles CSS à appliquer aux templates Keycloak sur les pages (login, registration, account ..Ect) et les emails.

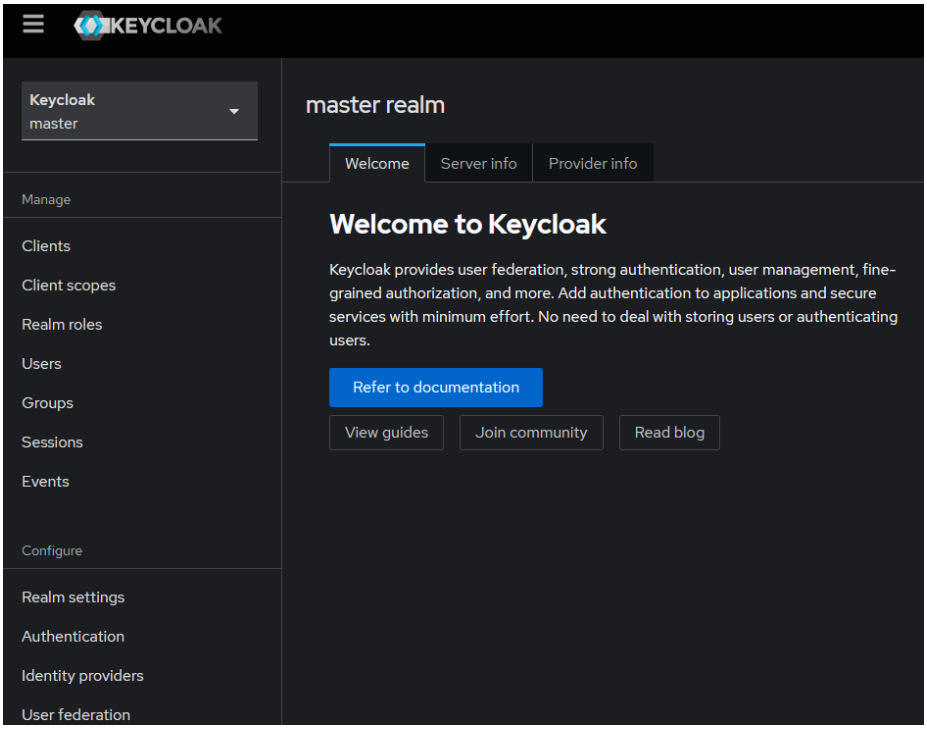
6

Configurer Keycloak server



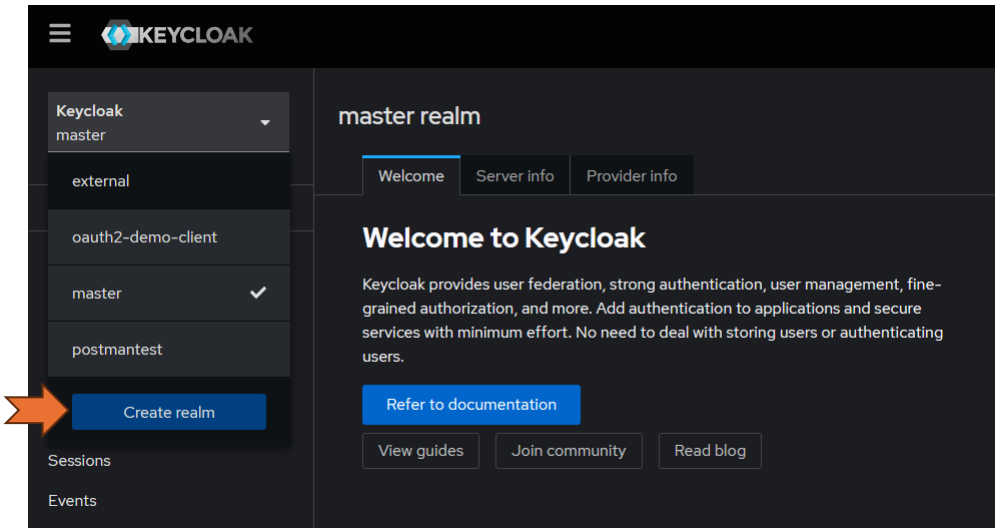
7

Configurer Keycloak server



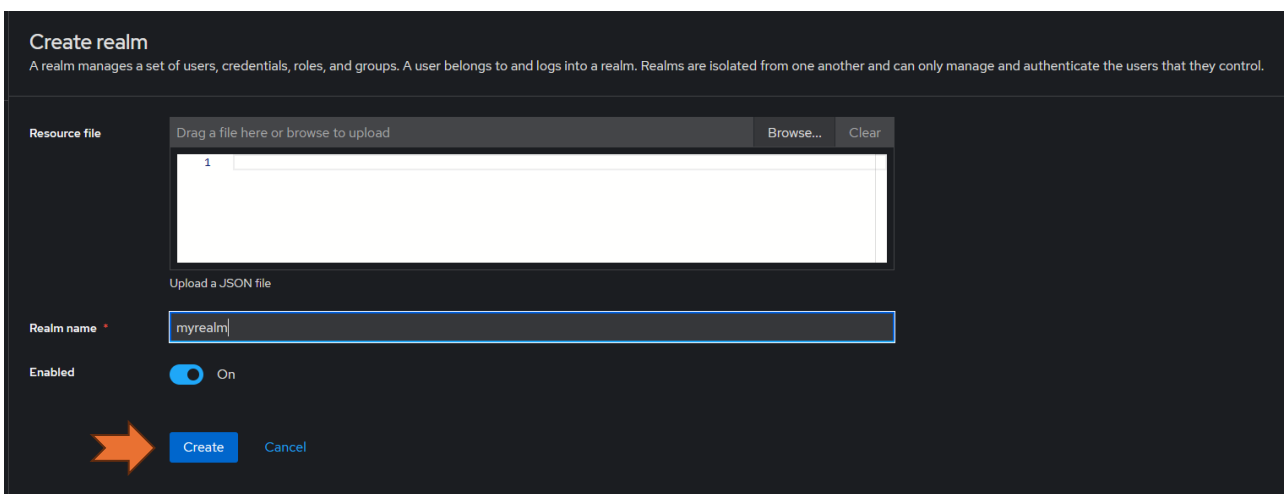
8

Configurer Keycloak server: Realm



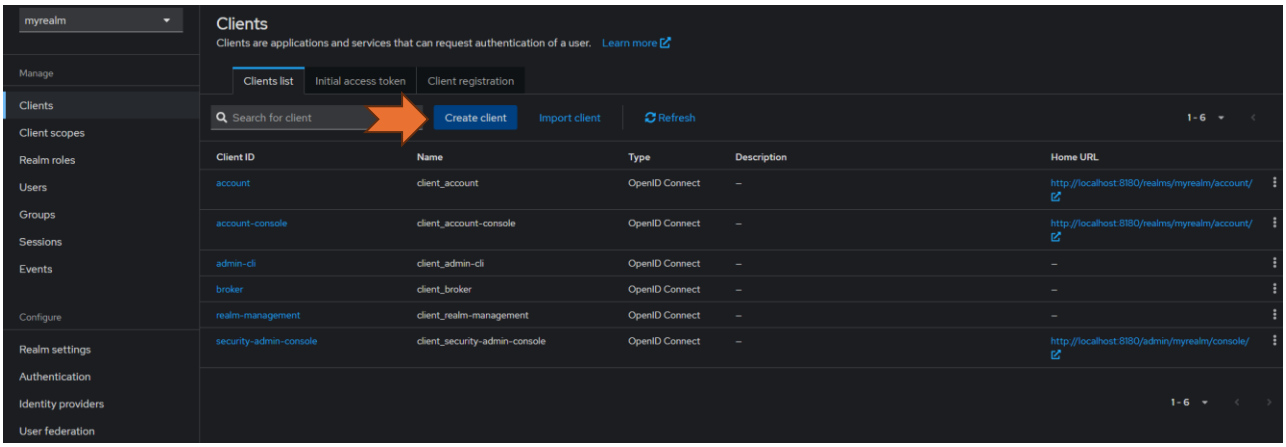
9

Configurer Keycloak server: Realm

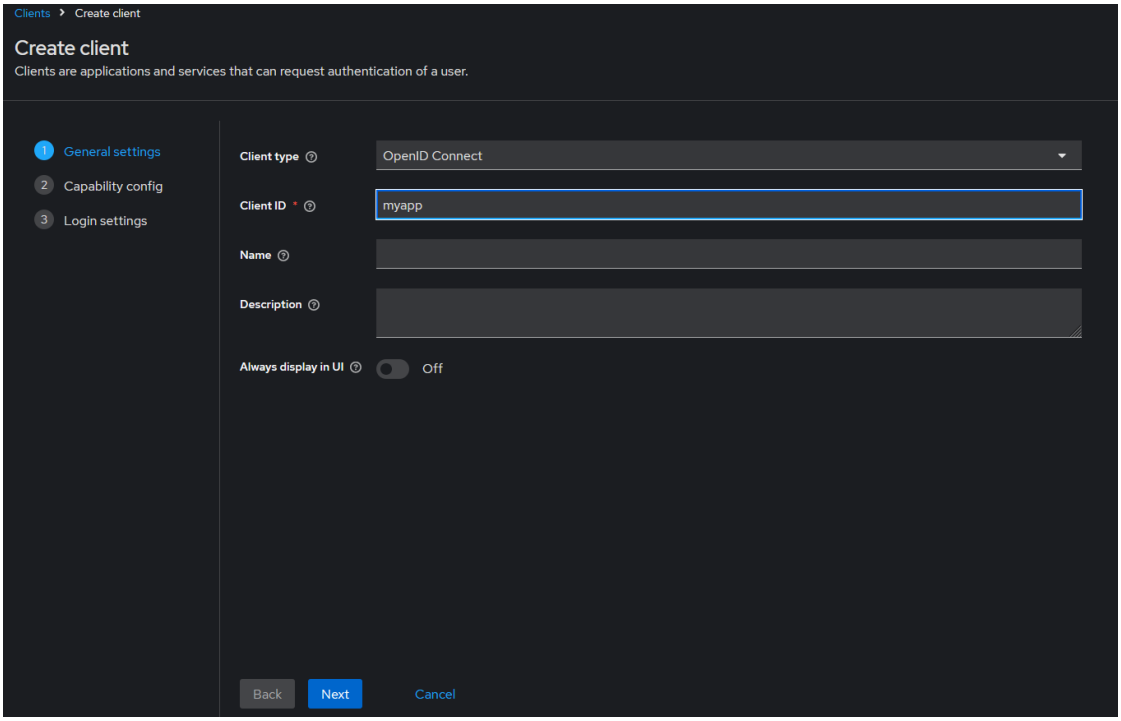


10

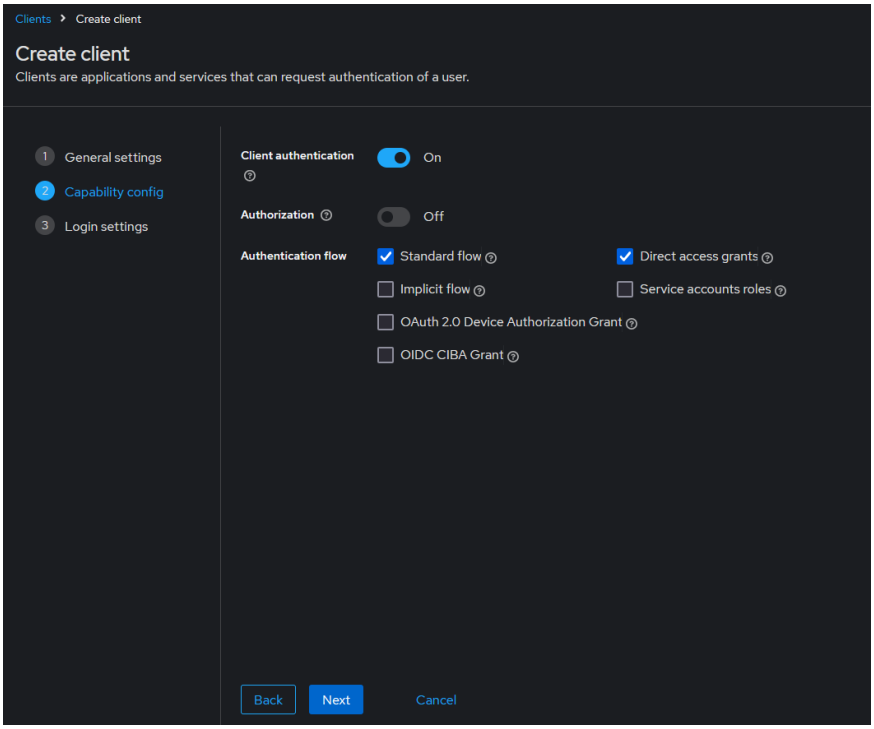
Créer un client : Ajouter l'application à sécuriser dans le realm



11

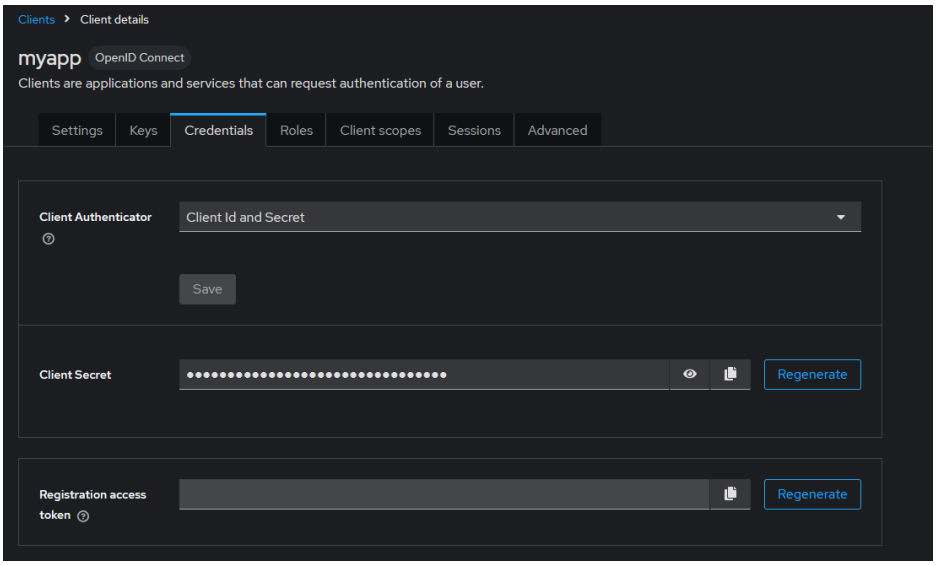


12



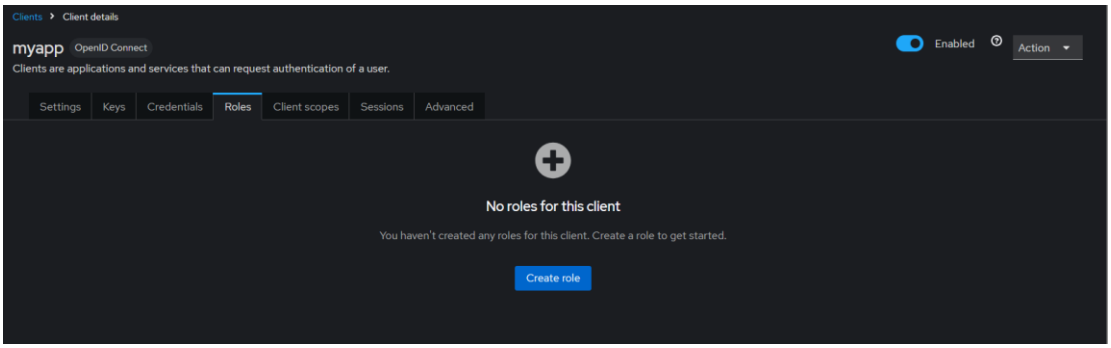
13

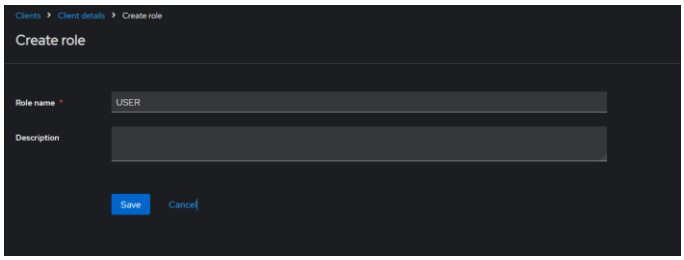
Pour récupérer le client secret



14

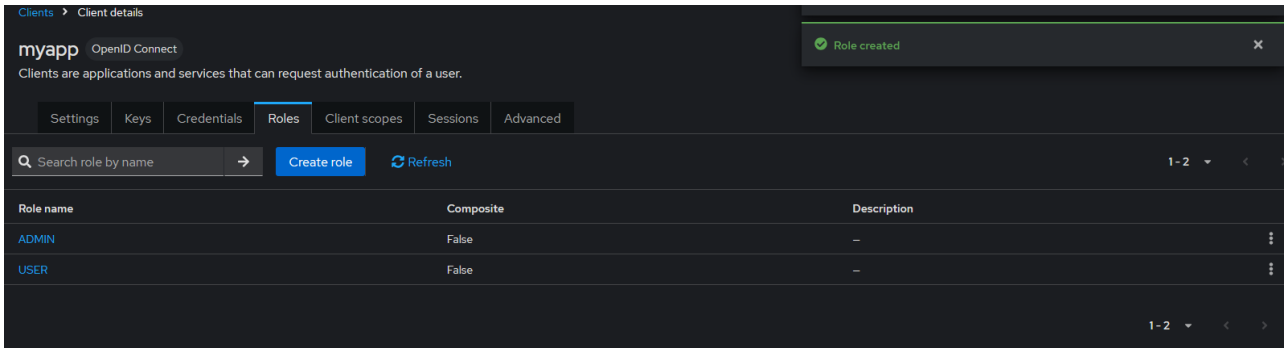
Créer les roles: USER et ADMIN





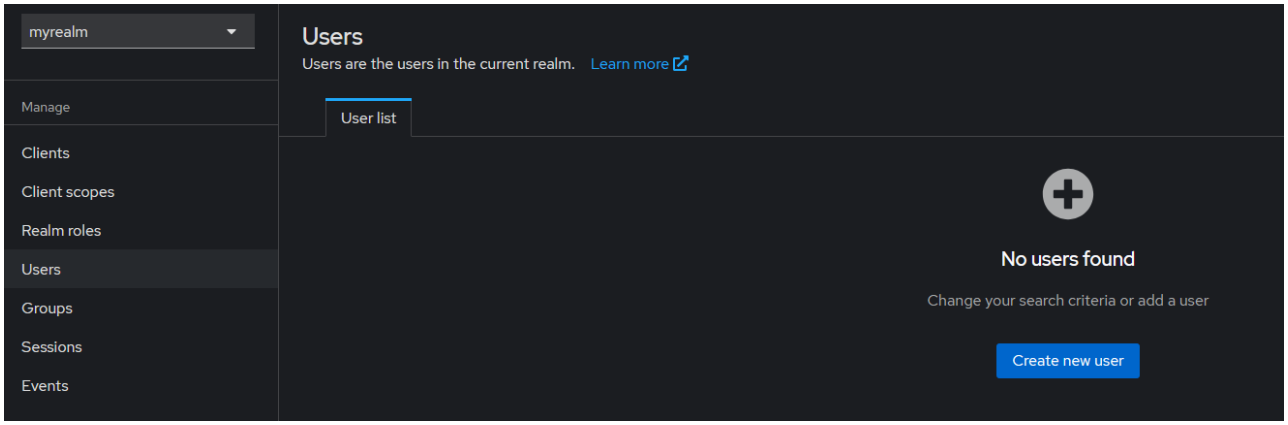
15

Créer les roles: USER et ADMIN



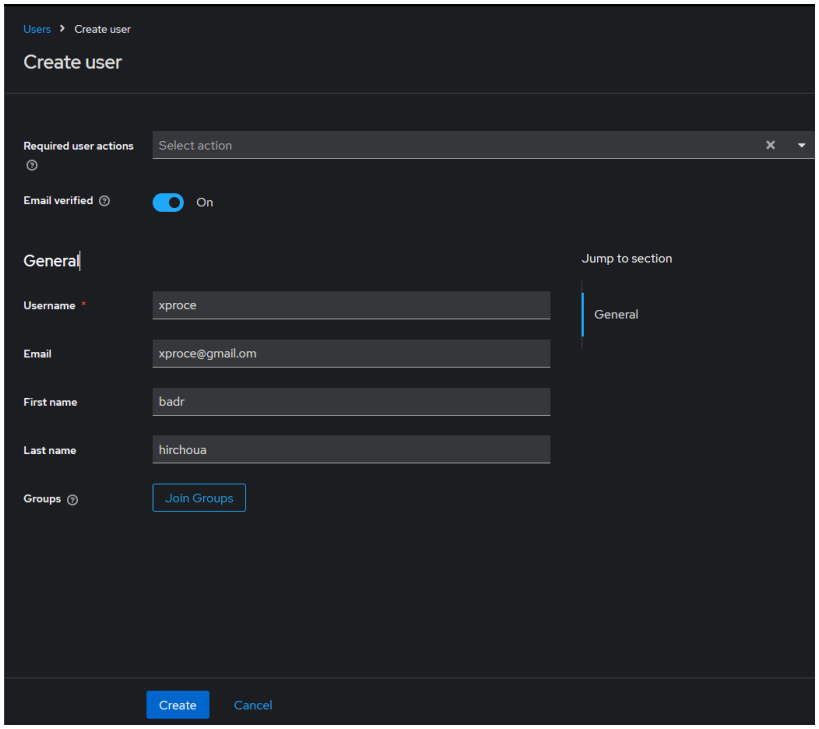
16

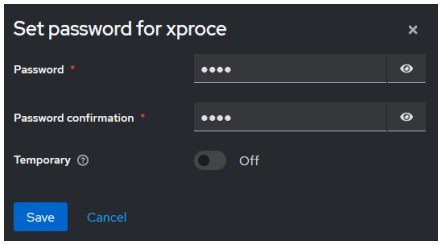
Créer un utilisateur



17

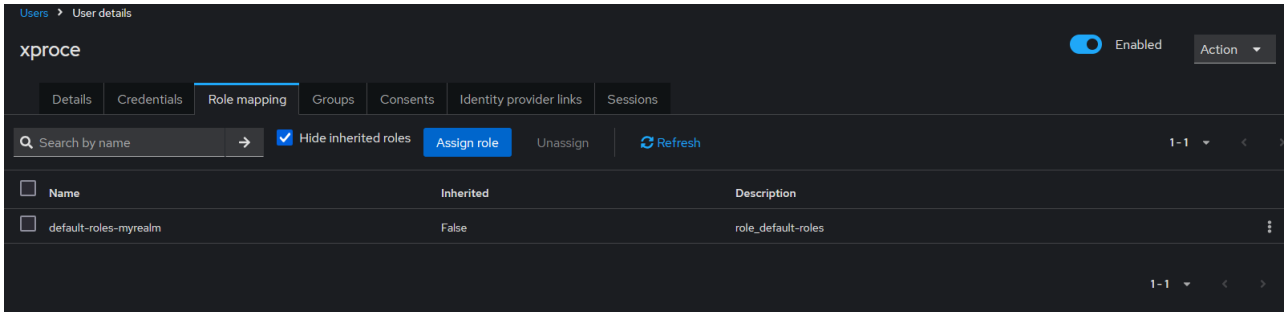
Créer un utilisateur





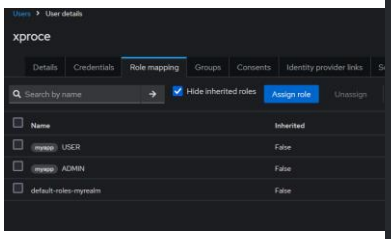
18

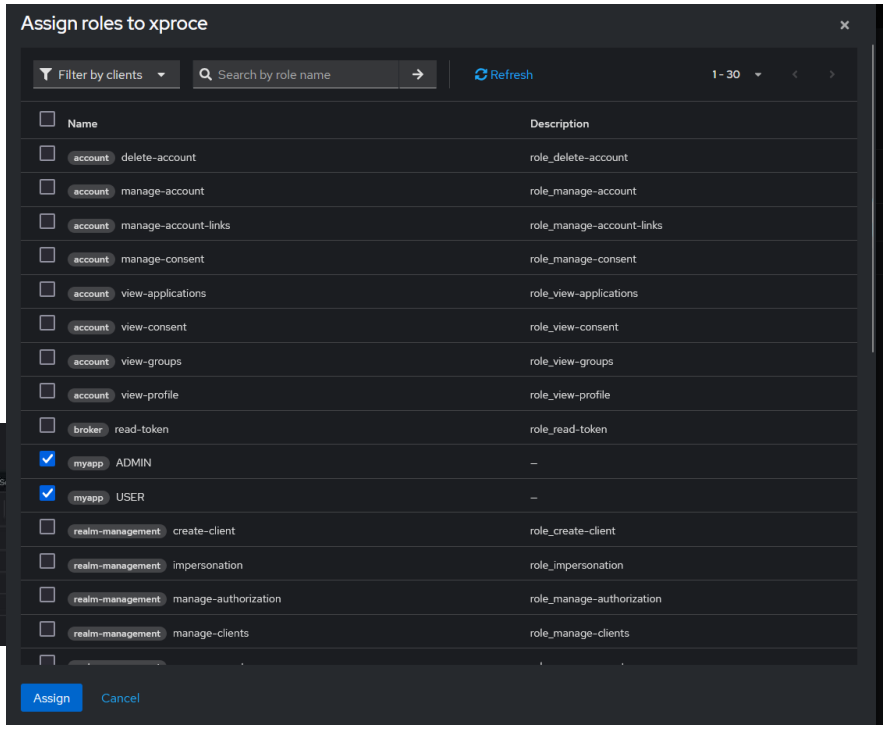
Créer un utilisateur: Ajouter Rôle(s)



19

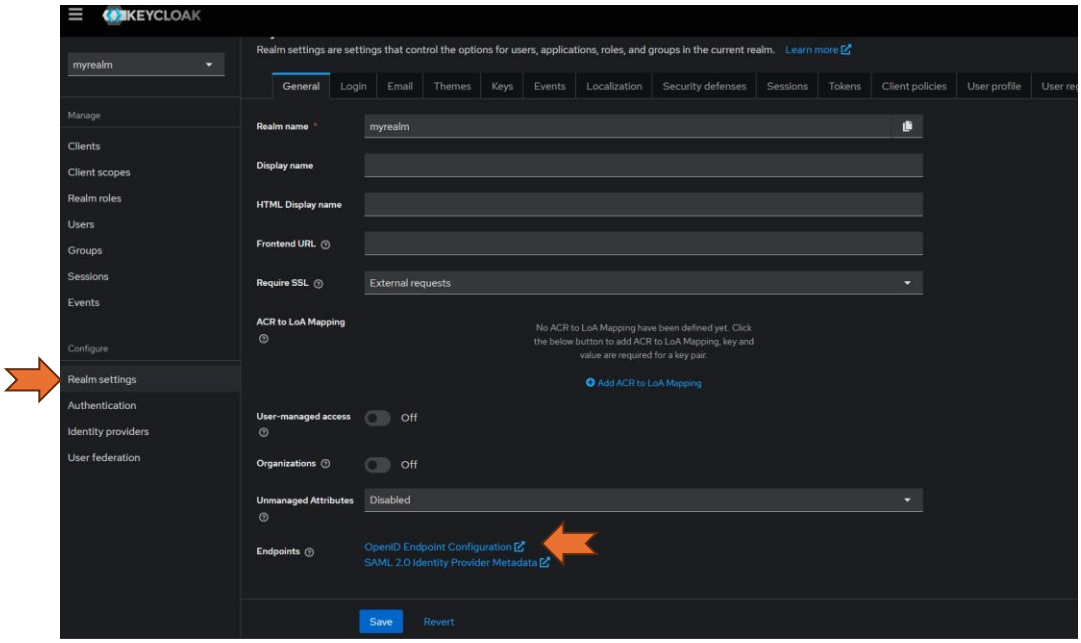
Créer un utilisateur: Ajouter Rôle(s)



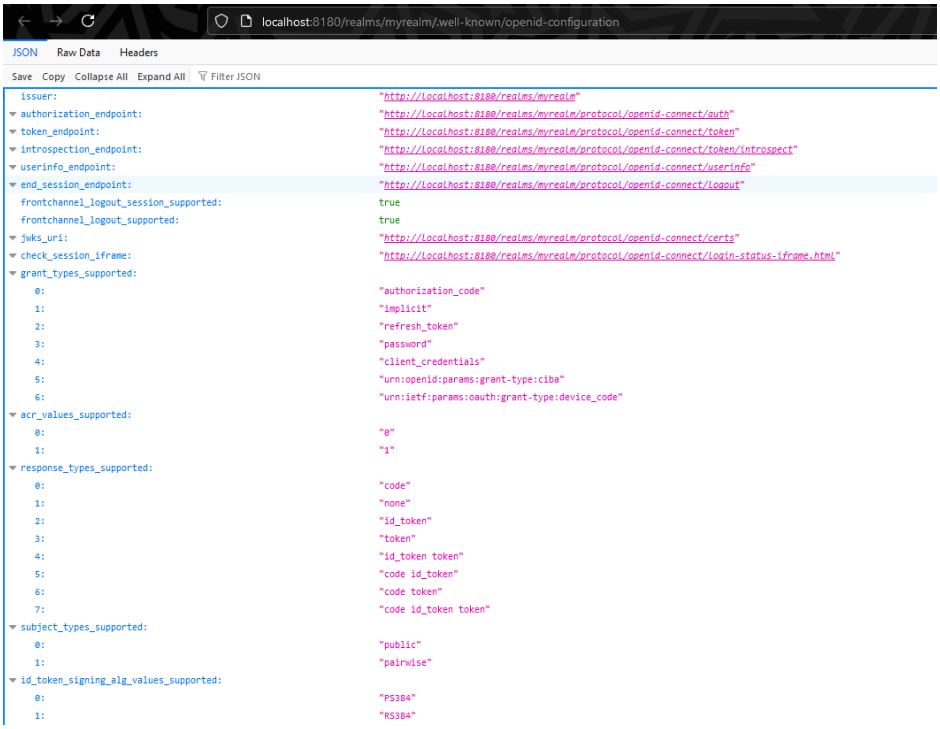


20

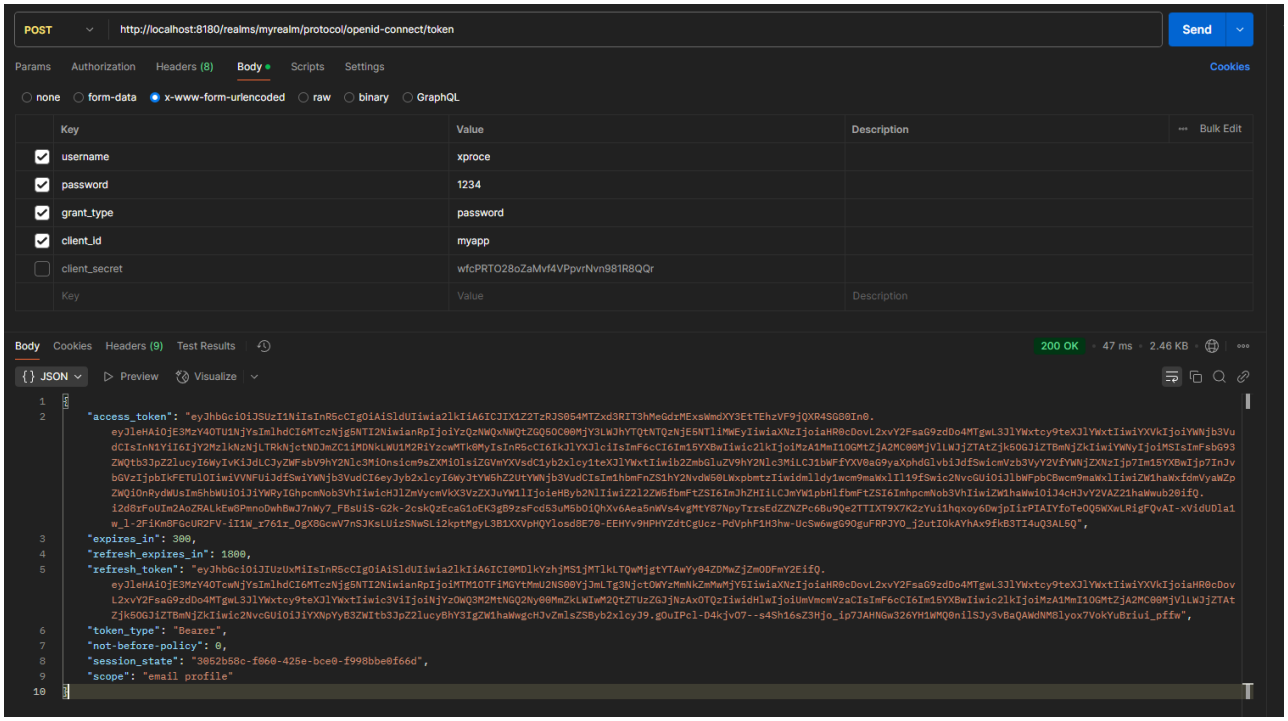
Tester avec Postman



21



22



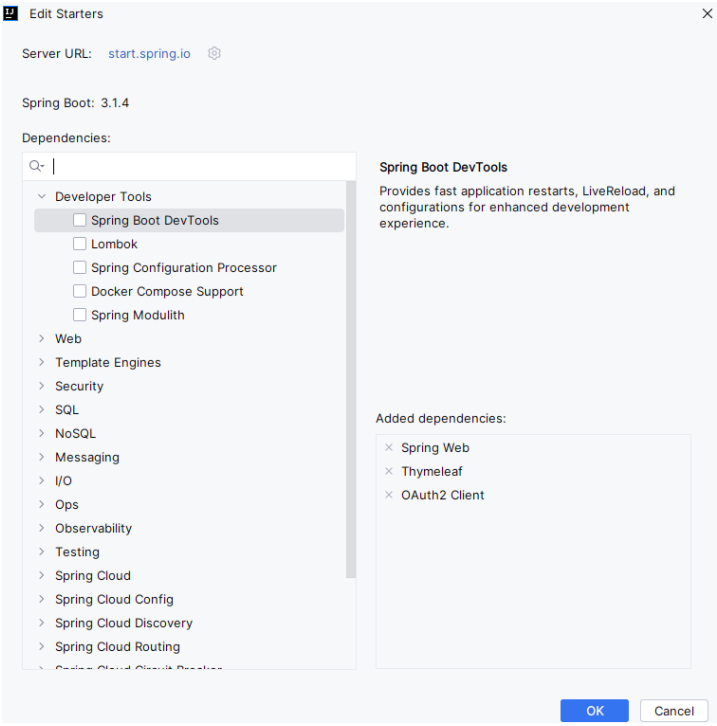
23

Exercice

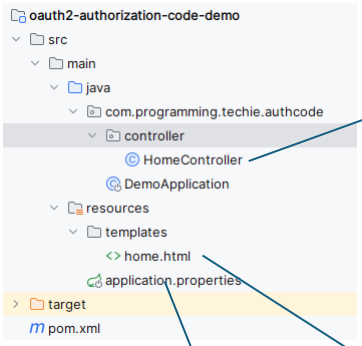
- Créer un Realm : oauth2-demo-client
- Créer un client: oauth2-demo-thymeleaf-client

24

Créer une application Spring boot avec les dépendances suivantes



25




```
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;

@Controller
public class HomeController {

   @GetMapping("/home")
   public String home(){
      return "home";
   }
}
```

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>Title</title>
</head>
<body>
<h1>Welcome to Oauth2 Spring MVC Demo App</h1>
</body>
</html>
```

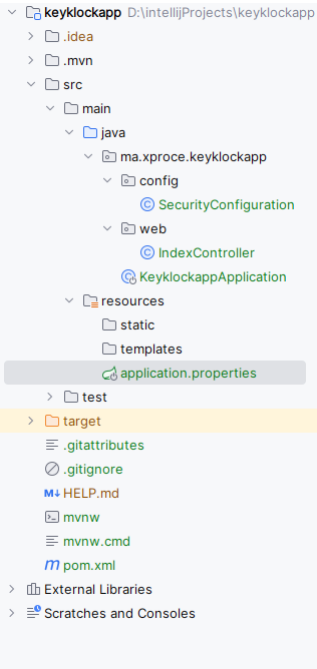
```
server.port=8089

spring.security.oauth2.client.registration.oauth2-demo-thymeleaf-client.client-id=oauth2-demo-thymeleaf-client
spring.security.oauth2.client.registration.oauth2-demo-thymeleaf-client.client-secret=OMGtCMD2RLWmNyBCTaFPW0DUBmHpeZb4
spring.security.oauth2.client.registration.oauth2-demo-thymeleaf-client.scope=openid, profile, roles
spring.security.oauth2.client.registration.oauth2-demo-thymeleaf-client.authorization-grant-type=authorization_code
spring.security.oauth2.client.registration.oauth2-demo-thymeleaf-client.redirect-uri=http://localhost:8089/login/oauth2/code/oauth2-demo-thymeleaf-client

spring.security.oauth2.client.provider.oauth2-demo-thymeleaf-client.issuer-uri=http://localhost:8180/realms/oauth2-demo-client
```

26

Exercice

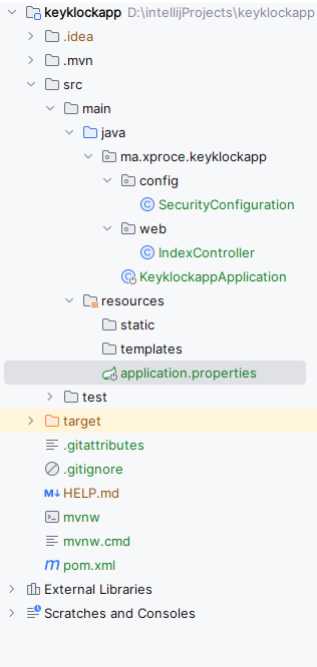
- Créer un Realm : external
- Créer un client: external-client

27



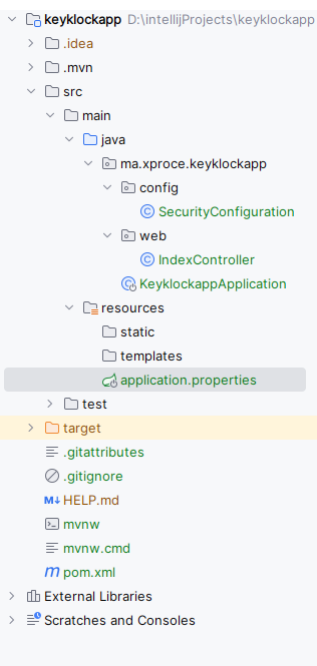

```
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.http.SessionCreationPolicy;
import org.springframework.security.web.SecurityFilterChain;

@Configuration
public class SecurityConfiguration {

  @Bean
  public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
    http
        .oauth2Client(httpSecurityOAuth2ClientConfigurer -> {})
        .oauth2Login(oauth2LoginConfigurer -> {
          oauth2LoginConfigurer.tokenEndpoint(tokenEndpointConfig -> {})
              .userInfoEndpoint(userInfoEndpointConfig -> {});
        });
    http
        .sessionManagement(sessionManagementConfigurer ->
            sessionManagementConfigurer.sessionCreationPolicy(SessionCreationPolicy.ALWAYS)
        );
    http
        .authorizeHttpRequests(authorizeHttpRequestsConfigurer ->
            authorizeHttpRequestsConfigurer
                .requestMatchers("/unauthenticated", "/oauth2/**", "/login/**").permitAll()
                .anyRequest().authenticated()
        )
        .logout(logoutConfigurer ->
            logoutConfigurer.logoutSuccessUrl("http://localhost:8180/realms/external/protocol/openid-
connect/logout?redirect_uri=http://localhost:8081/")
        );
    return http.build();
  }
}
```

28



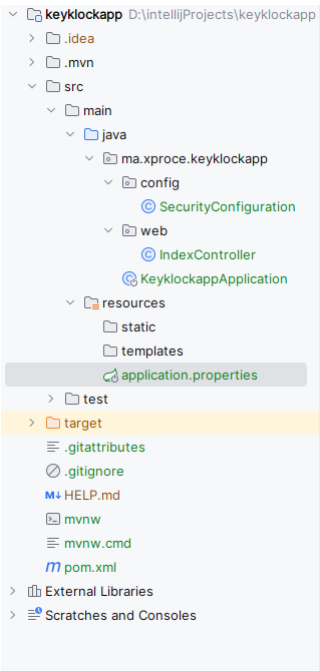

```java
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.security.oauth2.core.user.OAuth2User;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.security.core.GrantedAuthority;
import java.util.HashMap;
import java.util.Set;
import java.util.stream.Collectors;

@RestController
public class IndexController {
    @GetMapping(path = "/")
    public HashMap index() {

        OAuth2User user = ((OAuth2User) SecurityContextHolder.getContext().getAuthentication().getPrincipal());

        Set<String>  rolesSet = user.getAuthorities().stream()
                .map(GrantedAuthority::getAuthority)
                .collect(Collectors.toSet());
        return new HashMap() {{
            put("hello", user.getAttribute("name"));
            put("your email is: ", user.getAttribute("email"));
            put("your Authorities are : " , rolesSet);

        }};
    }

    @GetMapping(path = "/unauthenticated")
    public HashMap unauthenticatedRequests() {
        return new HashMap() {{
            put("this is ", "unauthenticated endpoint");
        }};
    }
}
```

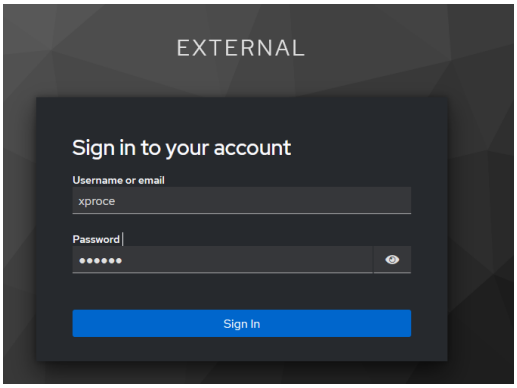
29

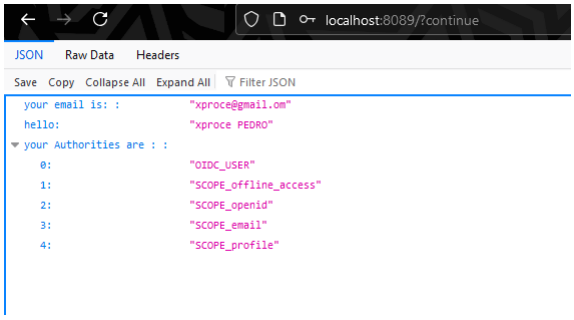



```yaml
spring.application.name=keyklockapp
server.port=8089
spring.security.oauth2.client.provider.external.issuer-uri=http://localhost:8180/realms/external
spring.security.oauth2.client.registration.external.provider=external
spring.security.oauth2.client.registration.external.client-name=external-client
spring.security.oauth2.client.registration.external.client-id=external-client
spring.security.oauth2.client.registration.external.client-secret=sXtCPCcfvQbQIP2pToN9sQb2A78Cv9RZ
spring.security.oauth2.client.registration.external.scope=openid,offline_access,profile
spring.security.oauth2.client.registration.external.authorization-grant-type=authorization_code
```

30



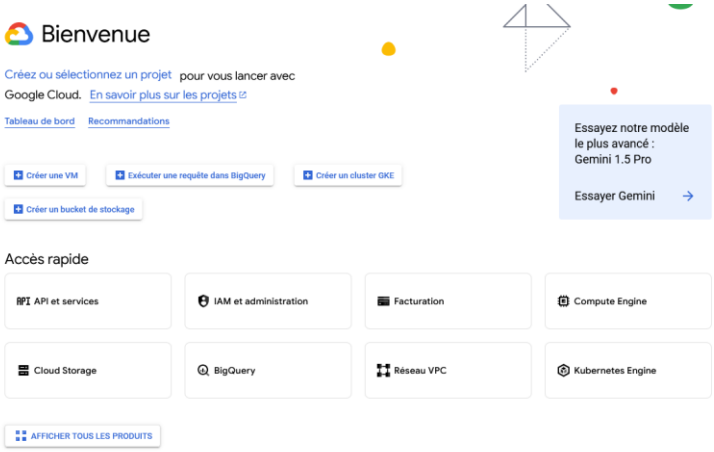




31

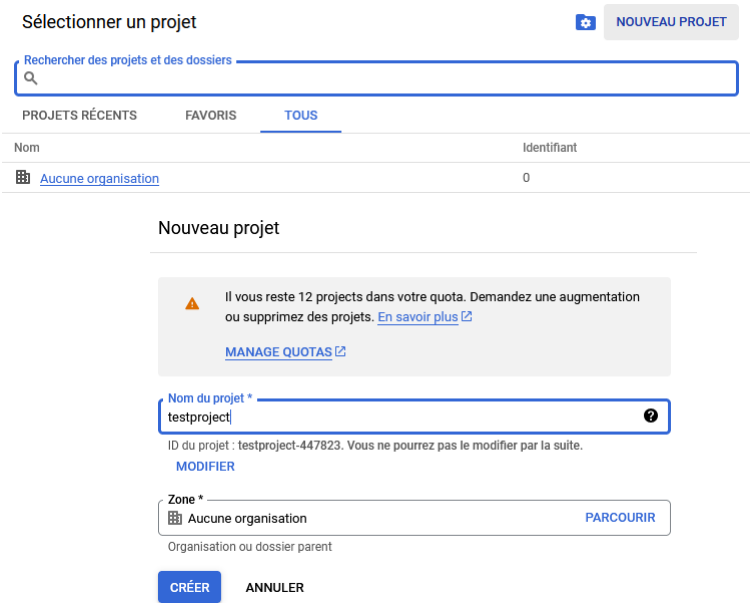
Creating OAuth2 client ID in Google cloud

- Log in to the Google Cloud Console. From the projects list, select a project or create a new one. If the APIs & Services page isn't already open, open the console's left-side menu and select APIs & Services. On the left, click Credentials, then “Create credentials,” and select OAuth client ID.



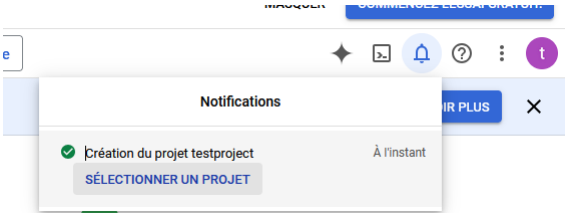
32

Creating OAuth2 client ID in Google cloud



33

Creating OAuth2 client ID in Google cloud



34

Sécurisez votre compte et protégez vos données en activant l'authentification multifacteur. Ajoutez un niveau de sécurité supplémentaire dès aujourd'hui.

API et services / Écran de consentement OAuth

- API et services activés
- Bibliothèque
- Identifiants
- Écran de consentement OAuth
- Page des accords d'utilisation

Écran de consentement OAuth

La gestion de l'écran de consentement OAuth évolue. Cette page a été remplacée pour offrir une expérience plus simple. Les pages actuelles deviendront indisponibles dans quelques jours.

DÉCOUVRIR LA NOUVELLE INTERFACE

Choisissez la manière dont vous souhaitez configurer et enregistrer votre application, y compris vos utilisateurs cibles. Vous ne pouvez associer qu'une application à votre projet.

User Type

Interne

Uniquement disponible pour les utilisateurs de votre organisation. Vous n'aurez pas besoin de faire valider votre application. En savoir plus sur le type d'utilisateur

Externe

Disponible pour tous les utilisateurs de test disposant d'un compte Google. Votre application démarrera en mode test et ne sera accessible qu'aux personnes figurant sur votre liste d'utilisateurs test. Une fois votre application prête pour la production, il est possible que vous deviez la faire valider. En savoir plus sur le type d'utilisateur

CRÉER

Donnez-nous votre avis sur votre expérience OAuth.

35

API et services / Écran de consentement OAuth / Modifier l'enregistrement de l'application

- API et services activés
- Bibliothèque
- Identifiants
- Écran de consentement OAuth
- Page des accords d'utilisation

Modifier l'enregistrement de l'application

La gestion de l'écran de consentement OAuth évolue. Cette page a été remplacée pour offrir une expérience plus simple. Les pages actuelles deviendront indisponibles dans quelques jours.

DÉCOUVRIR LA NOUVELLE INTERFACE

Informations sur l'application

Ces informations apparaissent dans l'écran de consentement, et permettent aux utilisateurs finaux de vous identifier et de vous contacter.

Nom de l'application *
myapp
Le nom de l'application demandant l'autorisation

Adresse e-mail d'assistance utilisateur *
testx@gmail.com
Permet aux utilisateurs de vous contacter s'ils ont des questions concernant leur consentement. En savoir plus

Logo de l'application

Ceci est votre logo. Il permet aux utilisateurs de reconnaître votre application et figure sur l'écran de consentement OAuth.
Après avoir importé un logo, vous devrez faire valider votre application, sauf si celle-ci est configurée uniquement pour une utilisation interne ou si son état de publication est "Test". En savoir plus

Affichage en cours du projet "testproj

36

API et services / Écran de consentement OAuth / Modifier l'enregistrement de l'application

- API et services activés
- Bibliothèque
- Identifiants
- Écran de consentement OAuth
- Page des accords d'utilisation

Modifier l'enregistrement de l'application

Écran de consentement OAuth — Niveaux d'accès — 3 **Utilisateurs tests** — 4 Résumé

La gestion de l'écran de consentement OAuth évolue. Cette page a été remplacée pour offrir une expérience plus simple. Les pages actuelles deviendront indisponibles dans quelques jours.

DÉCOUVRIR LA NOUVELLE INTERFACE

Utilisateurs tests

Quand l'état de publication est défini sur "Test", seuls les utilisateurs tests peuvent accéder à l'application. La limite d'utilisateurs autorisés avant la validation de l'application est de 100 pour toute sa durée de vie. En savoir plus

+ ADD USERS

Filtrer Saisissez le nom ou la valeur de la propriété

Informations utilisateur
Aucune ligne à afficher

ENREGISTRER ET CONTINUER ANNULER

37

Ajouter des utilisateurs

Quand l'état de publication est défini sur "Test", seuls les utilisateurs tests peuvent accéder à l'application. La limite d'utilisateurs autorisés avant la validation de l'application est de 100 pour toute sa durée de vie.

LEARN MORE

xproce@gmail.com

0 / 100

AJOUTER

38

API et services

API et services activés
Bibliothèque
Identifiants
Écran de consentement OAu...
Page des accords d'utilisation

Identifiants
+ CRÉER DES IDENTIFIANTS
SUPPRIMER
RESTAURER DES IDENTIFIANTS SUPPRIMÉS

Créez des identifiants pe

Clé API
Identifie votre projet à l'aide d'une clé API simple afin de vérifier le quota et l'accès

ID client OAuth
Demande à l'utilisateur d'autoriser l'application à accéder à ses informations

Compte de service
Active l'authentification de serveur à serveur au niveau de l'application à l'aide de comptes robots

Aidez-moi à choisir
Cet assistant pose quelques questions pour vous aider à choisir le type d'identifiant à utiliser

Clés API

Nom
Aucune clé API à affi

ID clients OAuth 2

Nom
Aucun client OAuth à afficher

Comptes de service

E-mail	Nom
Aucun compte de service à afficher	

39

Client OAuth créé

Vous pouvez toujours accéder à l'ID client et au code secret depuis la section "Identifiants" de la page "API et services"

L'accès OAuth est réservé aux utilisateurs de test listés sur votre écran de consentement OAuth

ID client	791939405877-20kvabf7s0t9gs0llc1ela6q scs636al.apps.googleusercontent.com
Code secret du client	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
Date de création	15 janvier 2025 à 00:33:08 GMT+1
État	Activé

TÉLÉCHARGER AU FORMAT JSON

API et services / Écran de consentement OAuth / Créer un ID client OAuth

API et services activés
Bibliothèque
Identifiants
Écran de consentement OAu...
Page des accords d'utilisation

Créer un ID client OAuth

OAuth 2.0. En savoir plus sur les types de clients OAuth

Type d'application *
Application Web

Nom *
Client Web 1

Nom de votre client OAuth 2.0. Ce nom ne sert qu'à identifier le client dans la console. Il n'est pas visible par les utilisateurs finaux.

Les domaines des URI que vous ajoutez ci-dessous seront automatiquement placés dans votre écran de consentementOAuth en tant que domaines autorisés.

Origines JavaScript autorisées

À utiliser avec les requêtes provenant d'un navigateur

URI 1 *
http://localhost:8090

+ AJOUTER UN URI

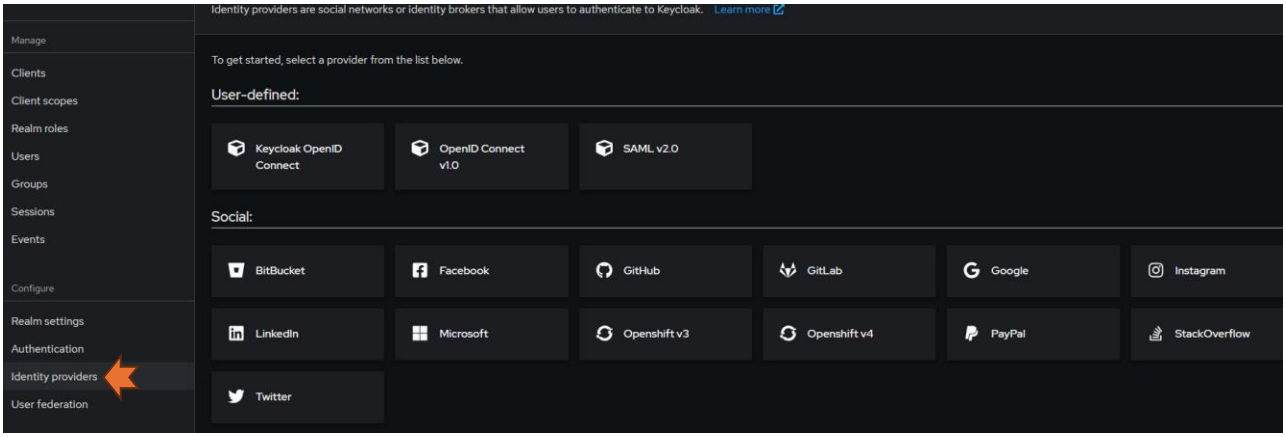
URI de redirection autorisés

À utiliser avec les requêtes provenant d'un serveur Web

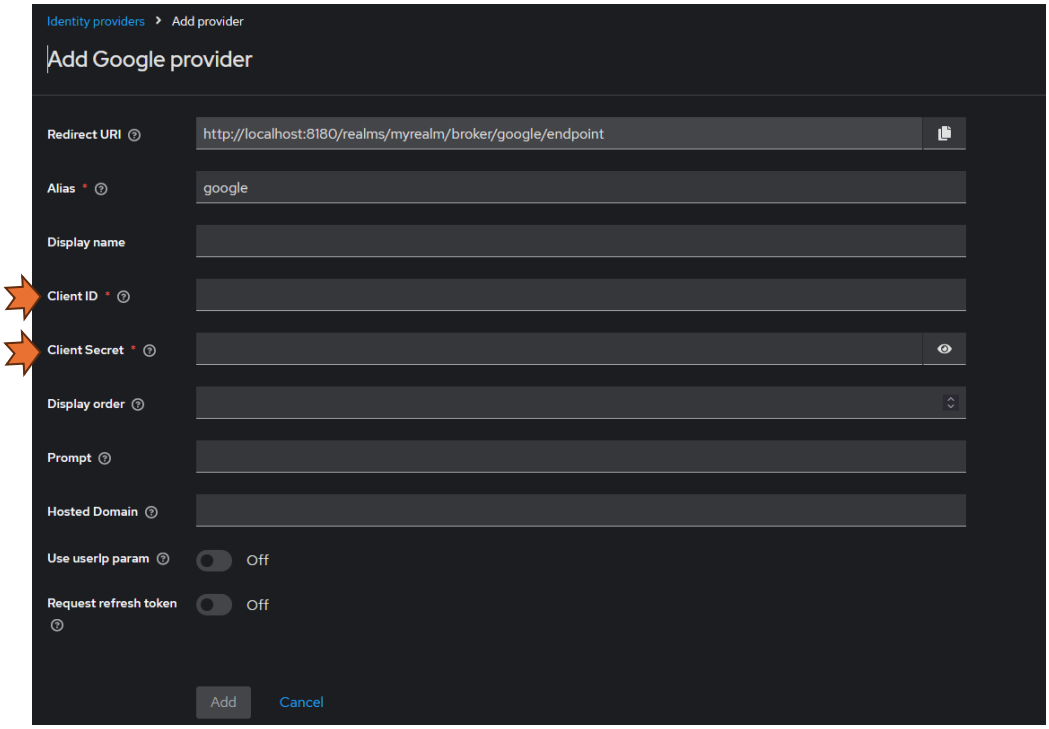
URI 1 *
http://localhost:8180/realms/oauth2-demo-client

+ AJOUTER UN URI

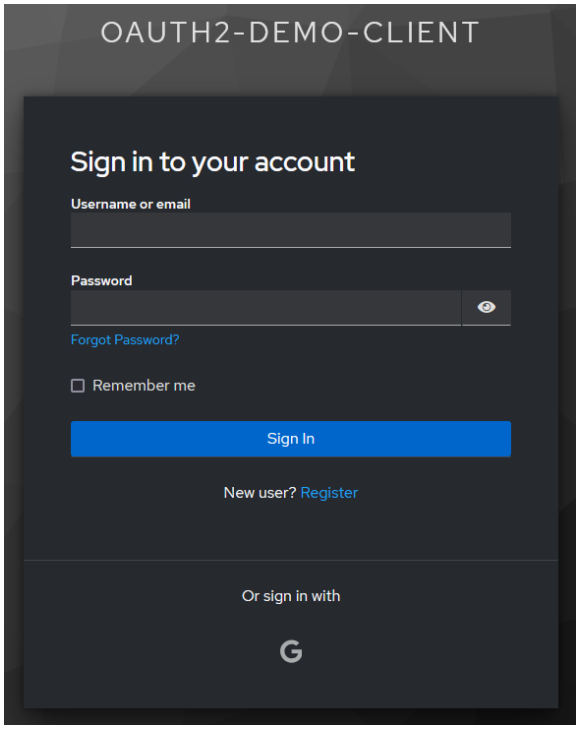
40


Identity providers are social networks or identity brokers that allow users to authenticate to Keycloak. Learn more
Manage
Clients
Client scopes
Realm roles
Users
Groups
Sessions
Events
Configure
Realm settings
Authentication
Identity providers
User federation
To get started, select a provider from the list below.
User-defined:
Keycloak OpenID Connect
OpenID Connect v1.0
SAML v2.0
Social:
BitBucket
Facebook
GitHub
GitLab
Google
Instagram
LinkedIn
Microsoft
Openshift v3
Openshift v4
PayPal
StackOverflow
Twitter

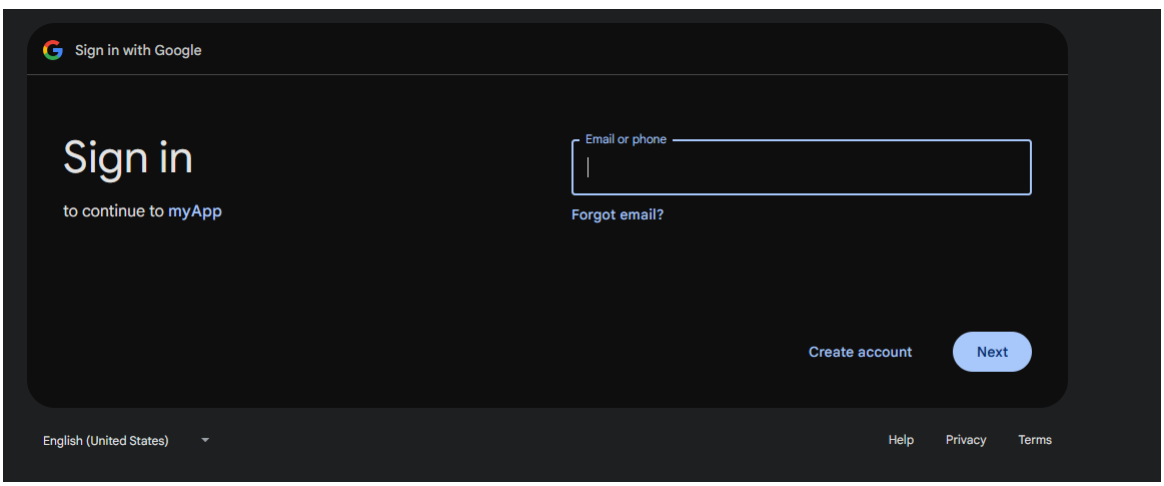
41


Identity providers > Add provider
Add Google provider
Redirect URI
http://localhost:8180/realms/myrealm/broker/google/endpoint
Alias *
google
Display name
Client ID *
Client Secret *
Display order
Prompt
Hosted Domain
Use userIp param
Off
Request refresh token
Off
Add
Cancel

42


OAUTH2-DEMO-CLIENT
Sign in to your account
Username or email
Password
Forgot Password?
Remember me
Sign In
New user? Register
Or sign in with

43


Sign in with Google
Sign in
to continue to myApp
Email or phone
Forgot email?
Create account
Next
English (United States)
Help
Privacy
Terms


localhost:8089/home
Welcome to Oauth2 Spring MVC Demo App

44