Spring Boot et OAuth

# Démonstration 1 du module 8

|  |
| --- |
| Les objectifs de cette démonstration sont   * Ajout des starters spring-boot-starter-security et spring-boot-starter-oauth2-client * Configuration avec le serveur OAuth2 : GitHub |

Contexte

* Nous continuons notre application précédente : demo-nosql.
  + Si votre application n’est pas complète, vous pouvez reprendre le code Java dans les ressources
  + Attention, le paramétrage du projet Spring Boot n’est pas inclus, il vous faut le compléter
* Dans cette itération, nous voulons utiliser les comptes GitHub pour s’autoriser l’accès à nos APIs

Déroulement

# Etape 1 : Dépendances Spring Boot :

Il faut ajouter les starters de Spring Security et d’OAuth2 client

* Dans build.gradle

//Security

**implementation** 'org.springframework.boot:spring-boot-starter-security'

**testImplementation** 'org.springframework.security:spring-security-test'

//OAuth2

**implementation** 'org.springframework.boot:spring-boot-starter-oauth2-client:3.1.3'

* Faire un Gradle Refresh Project

# Etape 2 : GitHub

L’idée est d’utiliser notre application au travers de GitHub et de ses autorisations.

Il faut donc déclarer notre application dans GitHub pour avoir le droit d’utiliser OAuth2

* Sous votre compte GitHub
* Allez sur vos paramètres
* Allez vers le bas sur l’option « Developer settings »

Une image contenant texte, capture d’écran, Police, blanc

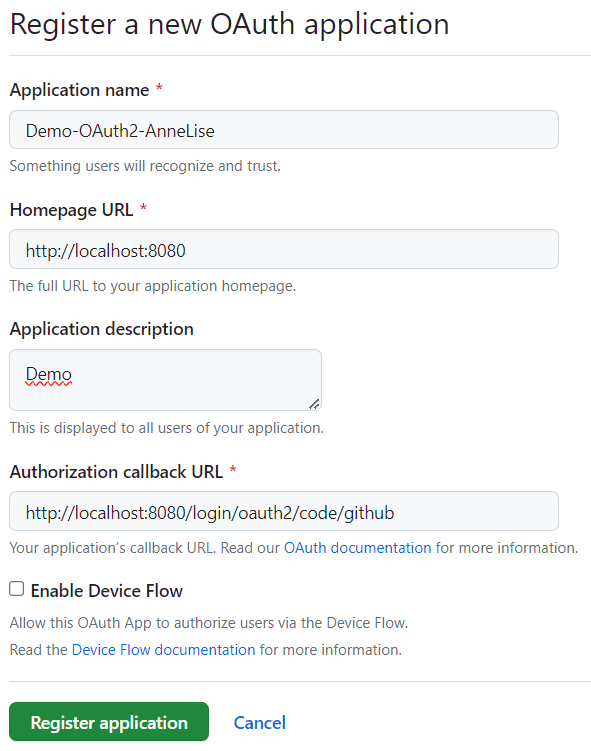
Description générée automatiquement

* Dans ce paramétrage, vous avez des choix d’autorisations :

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

* + Sélectionner OAuth Apps
* Donner un nom à votre application
  + Exemple : Demo-OAuth2-VotreNom
  + Indiquer un commentaire
  + Et l’url de la page d’accueil
    - Mettre l’URL de « localhost » : http://localhost:8080
  + L’URL d’autorisation 🡪 permet de gérer le jeton transmis par GitHub sur le contexte du projet (« localhost »)
    - http://localhost:8080/login/oauth2/code/github

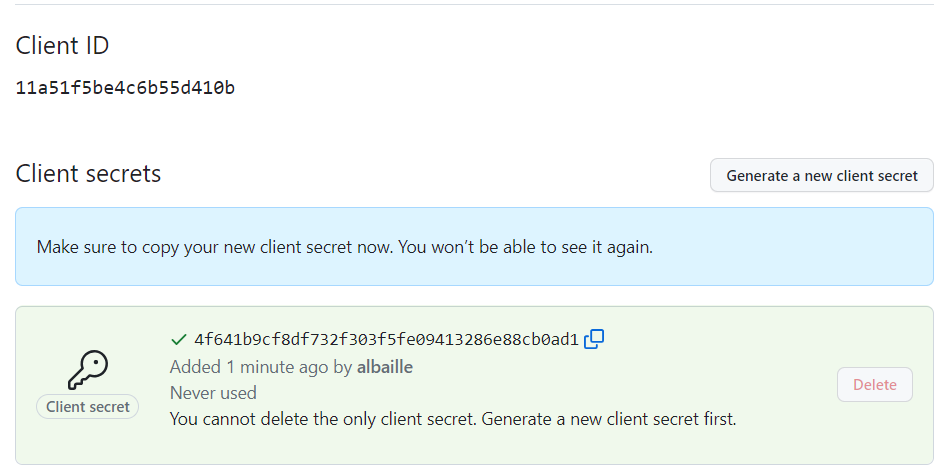


* Vous obtiendrez une vue récapitulative
  + Où, GitHub aura créé un identifiant Client ID :

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

* Client ID 🡪 copier le dans un bloc note il va nous servir plus tard
* Il nous faut générer un « Client Secret »
  + GitHub vous demandera de ressaisir votre mot de passe GitHub pour valider



* Il nous faut les 2 informations : Client ID et Client Secret pour effectuer toutes les opérations de connexion.

# Etape 3 : Application Spring Boot

* Ajouter les paramètres de OAuth 2.0 dans le fichier de configuration de Spring Boot : application.properties ou application.yml

#configuration OAuth2 - GitHub

spring.security.oauth2.client.registration.github.client-id=11a5…

spring.security.oauth2.client.registration.github.client-secret=4f64…

#configuration OAuth2 - GitHub

security:

oauth2:

client:

registration:

github:

client-id: 11a5…

client-secret: 4f64…

* Créer une classe de configuration de sécurité
  + Définir les URLs accessibles par tous
  + Définir les URLs pour lesquels il faut être autorisé et authentifié
  + Définir l’utilisation de OAuth 2.0 comme solution de login

**package** fr.eni.demo.security.oauth;

**import** org.springframework.context.annotation.\*;

**import** org.springframework.security.config.annotation.web.builders.HttpSecurity;

**import** org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

**import** org.springframework.security.web.SecurityFilterChain;

@Configuration

@EnableWebSecurity

**public** **class** SecurityConfig {

/\*\*

\* il faut être authentifié pour accéder à l'API

\*/

@Bean

SecurityFilterChain filterChain(HttpSecurity http) **throws** Exception {

http

.authorizeHttpRequests(auth -> {

auth

// Permettre l'accès à l'URL racine à tout le monde

.requestMatchers("/").permitAll()

// Il faut être connecté pour les autres URLs

.anyRequest().authenticated();

})

// Indiquer à Spring Security l'utilisation de OAuth 2.0

.oauth2Login(oauth -> {

});

**return** http.build();

}

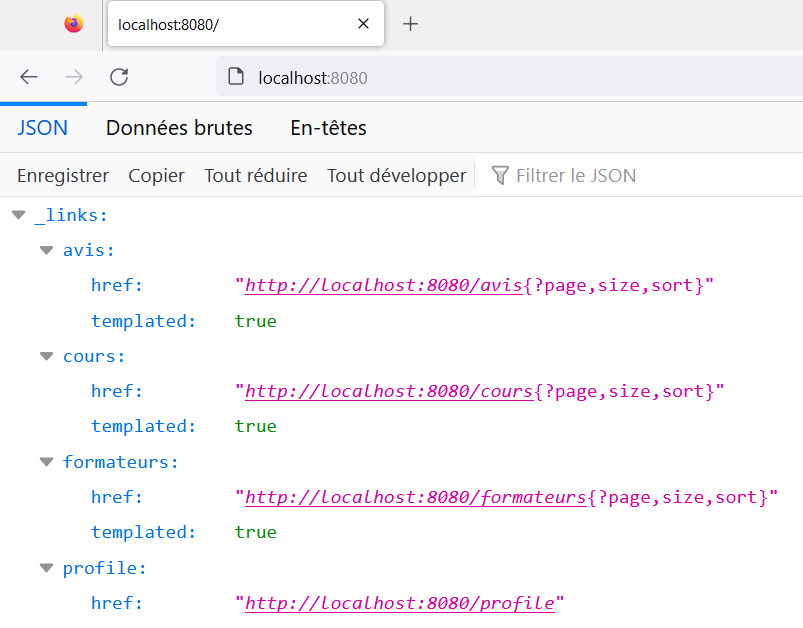
}

# Etape 4 : Exécution et autorisation GitHub

* Démarrer votre application

## URL racine : http://localhost:8080

* Essayer d’accéder à l’URL racine : http://localhost:8080
  + Tout le monde a le droit, vous aurez bien les liens principaux de Spring Data REST pour notre application :



## URL Swagger-UI : http://localhost:8080/eni/swagger-ui/index.html

* Essayer d’accéder à l’URL de Swagger-UI : http://localhost:8080/eni/swagger-ui/index.html
* 2 possibilités :
  1. Soit, vous avez un compte GitHub connecté
  2. Soit, vous n’êtes pas connecté

### Compte GitHub connecté

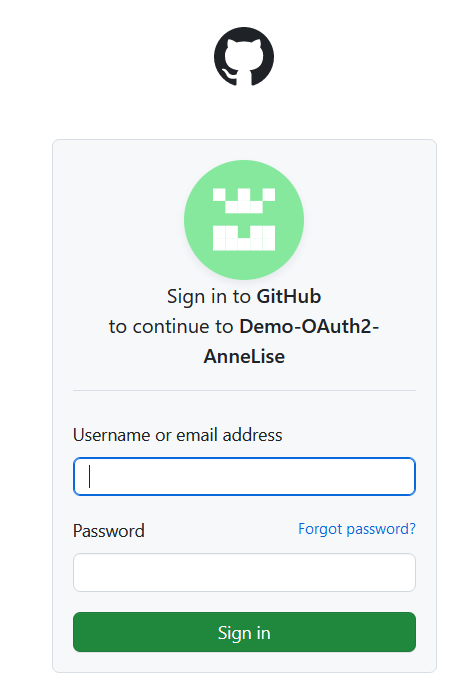
* Si vous vous placez dans un navigateur où vous êtes déjà connecté avec un compte GitHub
  + Vous serez redirigé vers GitHub pour demander l’autorisation de lire les données utilisateur pour la connexion par OAuth2 à notre application

Une image contenant texte, capture d’écran, Police, logo

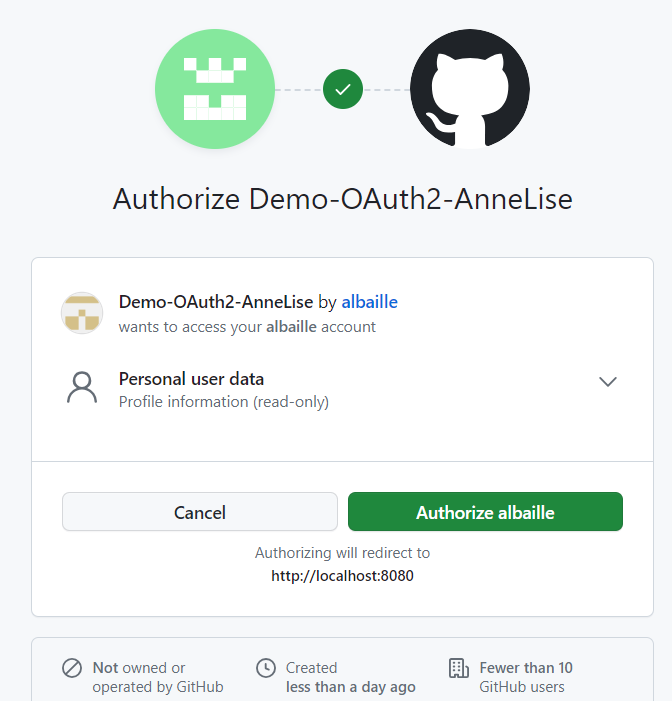
Description générée automatiquement

### Compte GitHub non connecté

* Si vous êtes dans un navigateur non connecté à GitHub
  + Spring vous redirigera vers une page de connexion GitHub pour notre application



* + Puis, vous aurez la demande d’autorisation de lire les données utilisateur pour la connexion par OAuth2

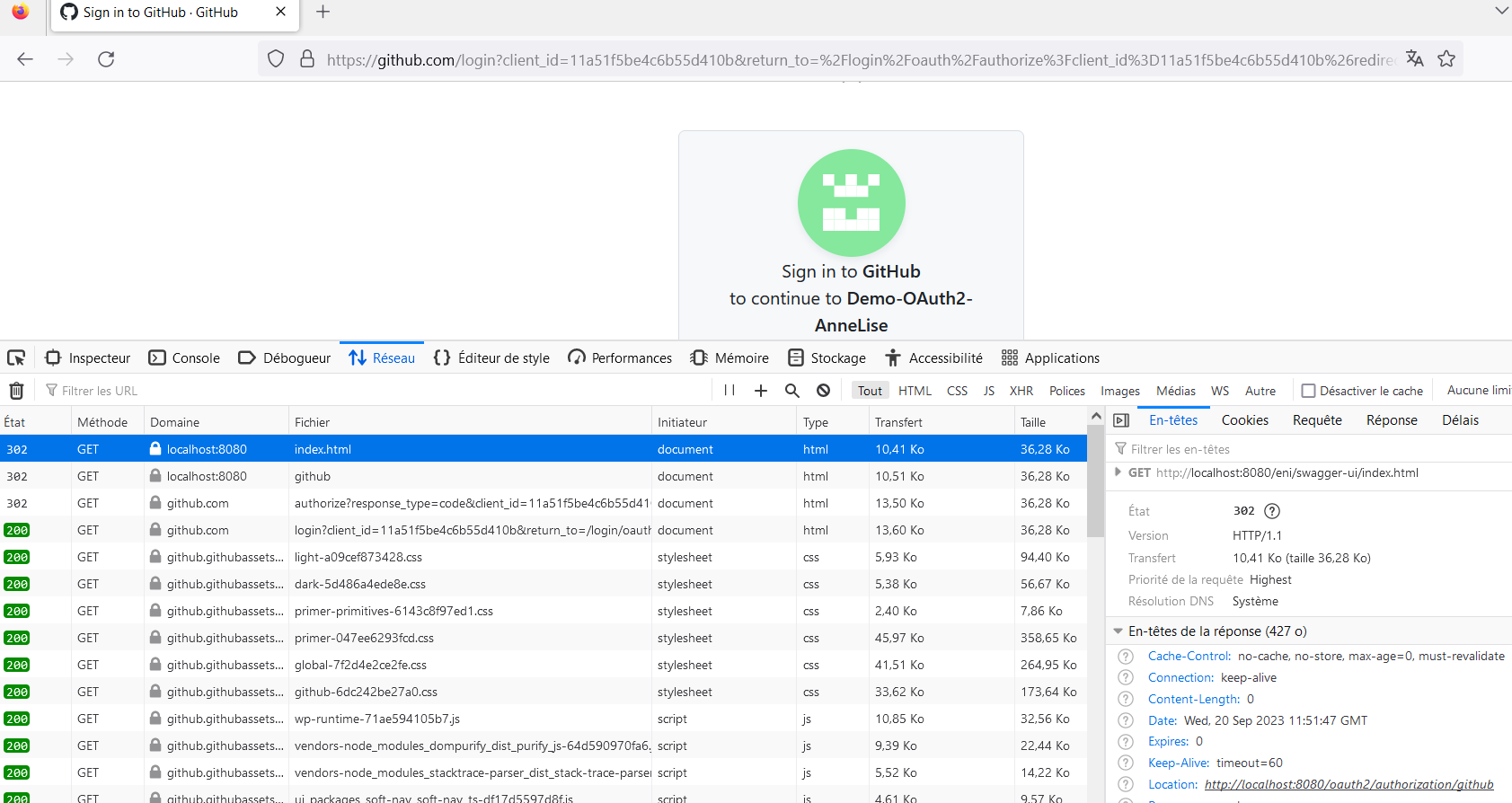


## Autorisation et cycle de vie du jeton de GitHub :

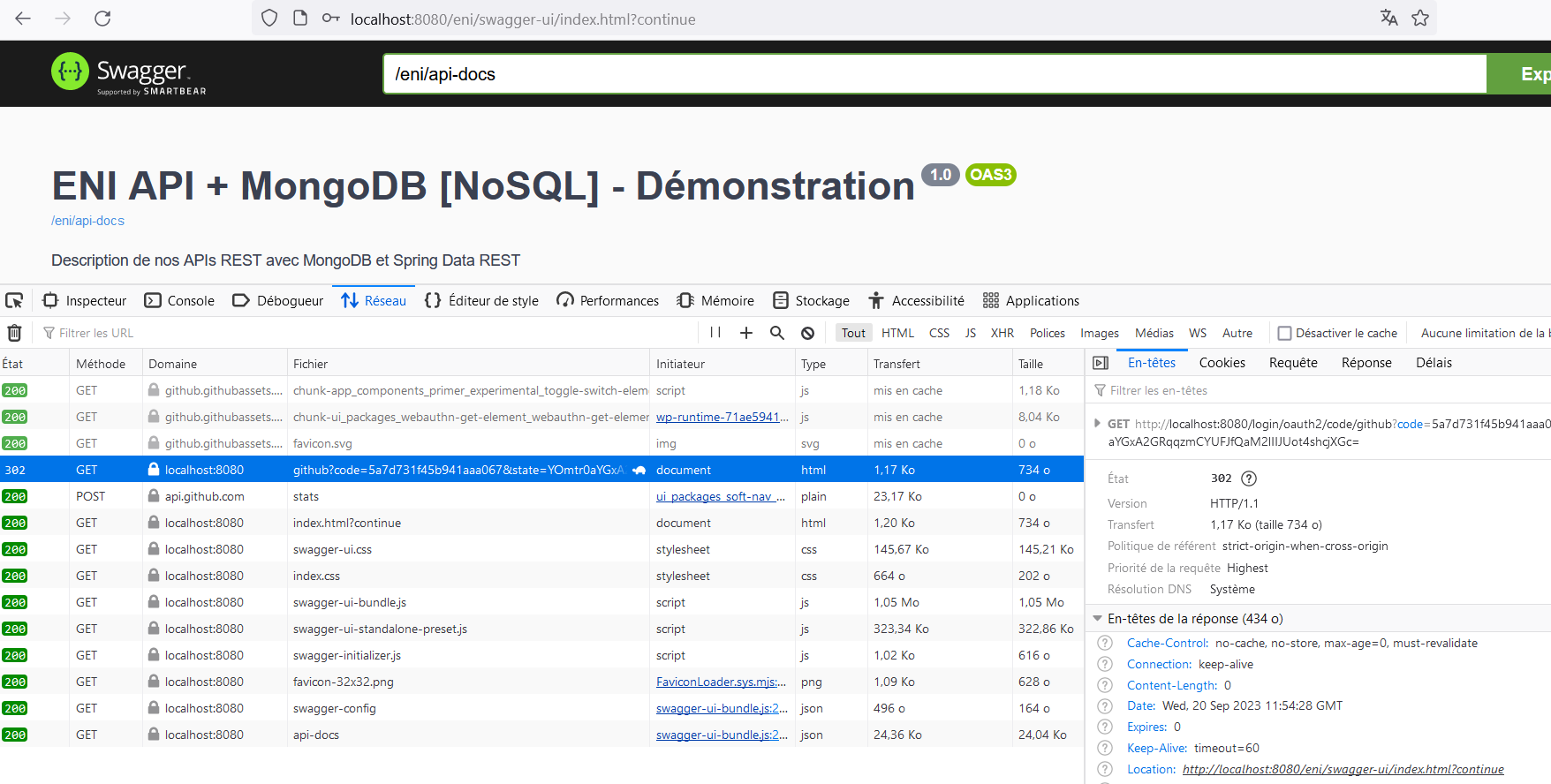
* Une fois que vous donnez votre accord pour l’autorisation de notre application d’accéder à votre compte GitHub en lecture
  + Vous serez redirigé vers Swagger-UI
  + Et Spring Security vous autorise à manipuler les APIs tant que vous êtes en session du navigateur et que votre jeton GitHub est actif

## Voir les étapes de l’autorisation via les outils de Dev sous le navigateur

* Pour montrer les étapes de l’autorisation : F12 dans le navigateur en mode privée
* Saisir l’url de Swagger-UI : http://localhost:8080/eni/swagger-ui/index.html
  + Avec aucun compte GitHub connecté



* Vous êtes redirigé, statut 302 vers l’url suivante : http://localhost:8080/oauth2/authorization/github
  + Cette url correspond à une demande d’autorisation 🡪 comme aucun utilisateur GitHub validé
  + Redirection vers l’URL : https://github.com/login/oauth/authorize?response\_type=...
  + Pour faire une demande d’autorisation sur l’application avec votre compte GitHub
* Si vous vous connectez
  + Et que vous acceptez de laisser notre application lire vos données :



* Vous aurez une session d’activée sous GitHub : https://github.com/session
  + Et vous aurez les autorisations de l’utilisateur
  + Vous serez donc redirigé vers http://localhost:8080/eni/swagger-ui/index.html
  + Et pour chaque requête ; il y aura une demande d’autorisation pour ce jeton de faite