

Documentation du Projet Microservice - Intégration de Apache Kafka et Keycloak

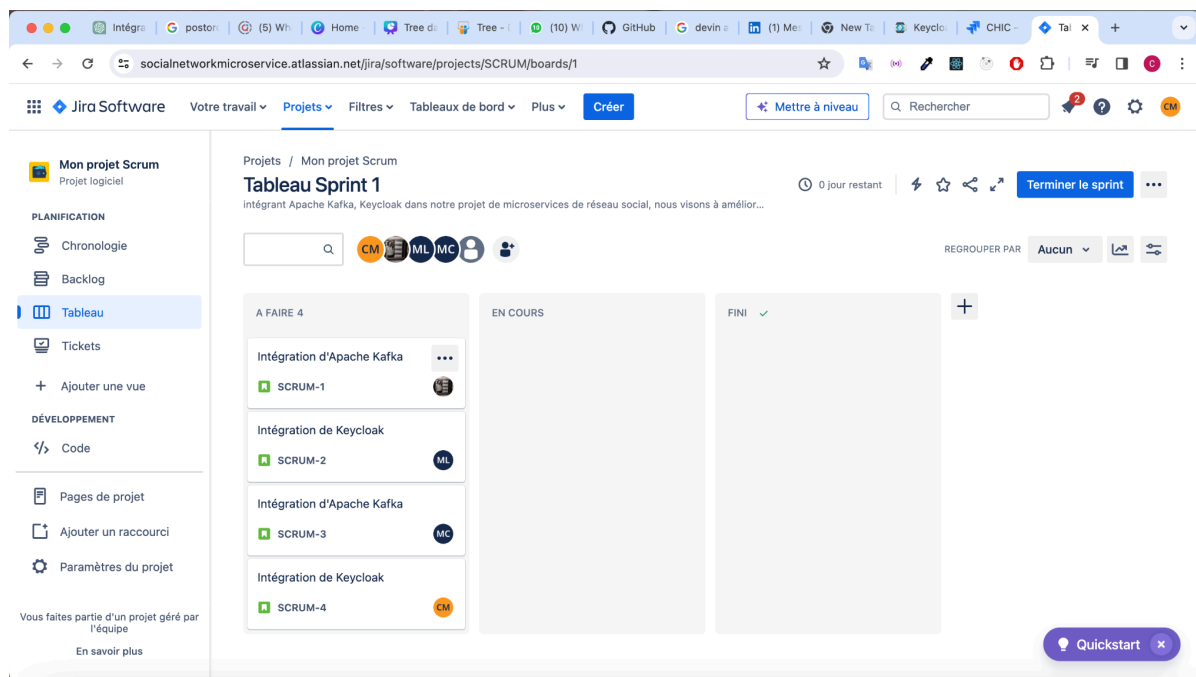
Introduction :

Les réseaux sociaux modernes nécessitent une architecture robuste et évolutive pour répondre aux exigences de scalabilité, de sécurité et de réactivité. Dans le cadre du projet social-network, nous cherchons à intégrer Apache Kafka et Keycloak pour améliorer ces aspects cruciaux de notre application de réseau social basée sur microservices.

Outil de Planification - Jira:



Jira est un outil de gestion de projet qui a été au cœur de notre processus de planification. Il a permis une gestion agile des sprints, des tâches, et un suivi en temps réel de l'avancement. La capture d'écran ci-dessous illustre notre tableau de planification dans Jira, mettant en évidence les sprints, les tâches assignées et les étapes de développement.



Intégration de Apache Kafka dans le Microservice de Notification de Social Network

Dans le cadre du projet Social Network, nous avons choisi d'intégrer Apache Kafka dans le microservice de Notification afin d'améliorer la réactivité et la fiabilité de notre système de gestion des notifications. Cette intégration permettra une communication asynchrone entre les différents composants de notre architecture de microservices, renforçant ainsi l'efficacité de notre application de réseau social moderne.

Fonctionnalités du Microservice de Notification :

Le microservice de Notification est responsable de la gestion et de la diffusion des notifications aux utilisateurs de Social Network. Il traite un grand volume d'événements générés par d'autres microservices, tels que les réactions aux publications, les nouveaux abonnements, ou les interactions avec les contenus.

Intégration d'Apache Kafka :

Utilisation de Kafka pour la diffusion des notifications :

Nous utilisons Apache Kafka comme canal de diffusion des notifications au sein de notre architecture. Chaque événement généré par les microservices est publié sur un sujet Kafka spécifique, permettant ainsi au microservice de Notification de les consommer de manière asynchrone.

Avantages de l'intégration de Kafka dans le Microservice de Notification :

- *Réactivité améliorée :* En adoptant une architecture basée sur des messages avec Kafka, le microservice de Notification peut traiter les événements de manière asynchrone, garantissant ainsi une réactivité accrue du système.
- *Fiabilité accrue :* La nature distribuée de Kafka assure une gestion robuste des événements, garantissant leur traitement même en cas de panne ou de surcharge du système.

- *Extensibilité* : L'utilisation de Kafka permet d'ajouter facilement de nouveaux types de notifications ou de modifier les règles de diffusion sans impacter les autres composants de l'application.

Intégration de Keycloak dans Social Network :



La sécurité des données utilisateur demeure une priorité essentielle au sein de notre application de réseau social, Social Network. En intégrant Keycloak, une solution de gestion des identités et des accès (IAM), nous visons à renforcer la sécurité de social network en mettant en œuvre des mécanismes avancés d'authentification et d'autorisation.

Fonctionnalités de Keycloak dans Social Network :

Authentification centralisée : Keycloak offre une plateforme centrale pour l'authentification des utilisateurs. Cette centralisation facilite grandement la gestion des identités au sein de notre architecture de microservices, garantissant ainsi une expérience utilisateur transparente et sécurisée.

Gestion des accès : Avec Keycloak, nous avons la capacité de mettre en place des politiques d'autorisation granulaires pour contrôler l'accès aux différentes ressources de notre application. Cette approche nous permet de garantir que seuls les utilisateurs autorisés peuvent accéder aux fonctionnalités spécifiques de Social Network, assurant ainsi la confidentialité et la sécurité des données.

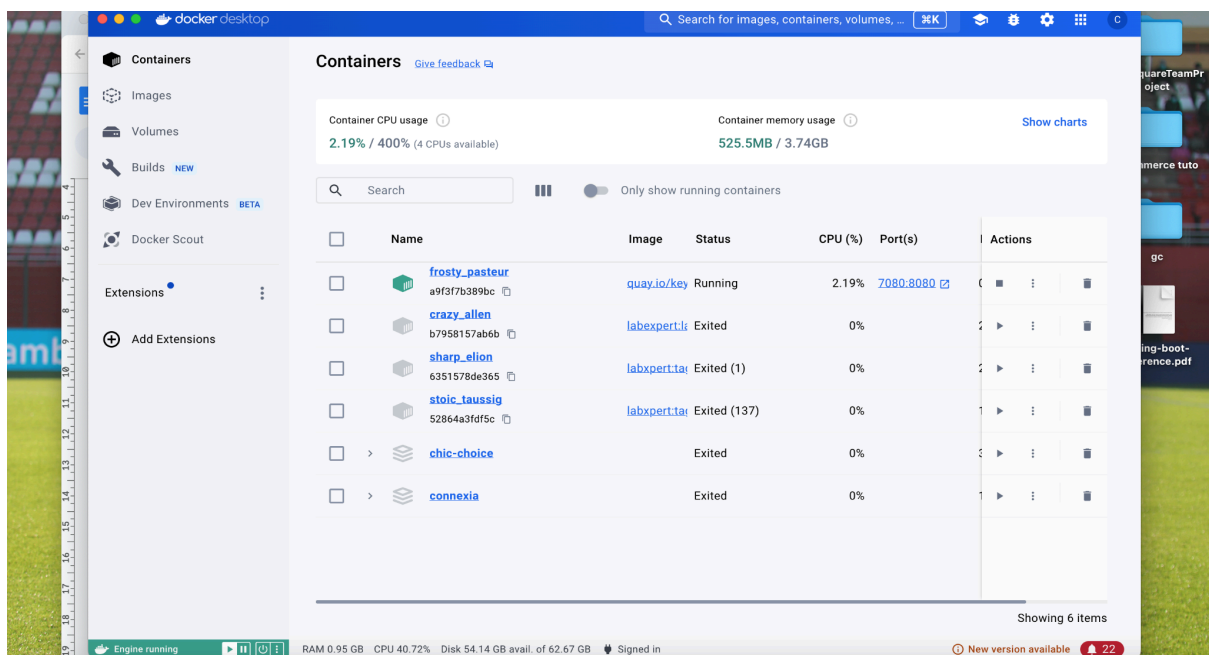
Sécurité renforcée : En utilisant les fonctionnalités avancées de Keycloak telles que l'authentification à deux facteurs et la gestion des sessions, nous renforçons la sécurité de Social Network contre les menaces potentielles. Ces mesures de sécurité supplémentaires permettent de protéger les comptes

utilisateur contre l'accès non autorisé et les tentatives d'intrusion.

Pour intégrer Keycloak dans notre projet Social Network, nous avons suivi les étapes suivantes :

Installation de Keycloak avec Docker :

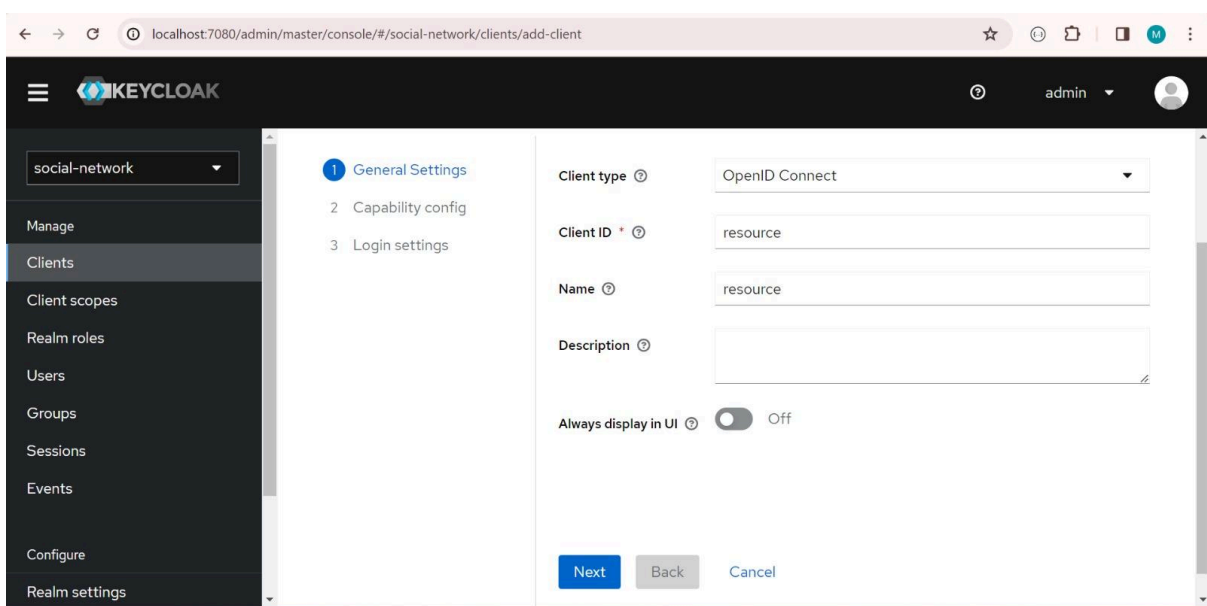
- Nous avons utilisé Docker pour simplifier le processus d'installation de Keycloak.
- Nous avons téléchargé l'image Keycloak depuis le référentiel Docker officiel en utilisant la commande `docker pull jboss/keycloak`.
- Ensuite, nous avons exécuté un conteneur Keycloak en utilisant la commande `docker run -p7080:8080 -e KEYCLOAK_USER=admin -e KEYCLOAK_PASSWORD=admin jboss/keycloak`.
- Cela a démarré un serveur Keycloak avec un utilisateur administrateur préconfiguré pour faciliter l'accès à l'interface d'administration.



Configuration du realm et du client :

- Après avoir accédé à l'interface d'administration de Keycloak via `http://localhost:8080/auth/admin`, nous avons créé un nouveau realm en cliquant sur "Add Realm".
- Nous avons donné un nom significatif à notre realm, par exemple "SocialNetworkRealm".

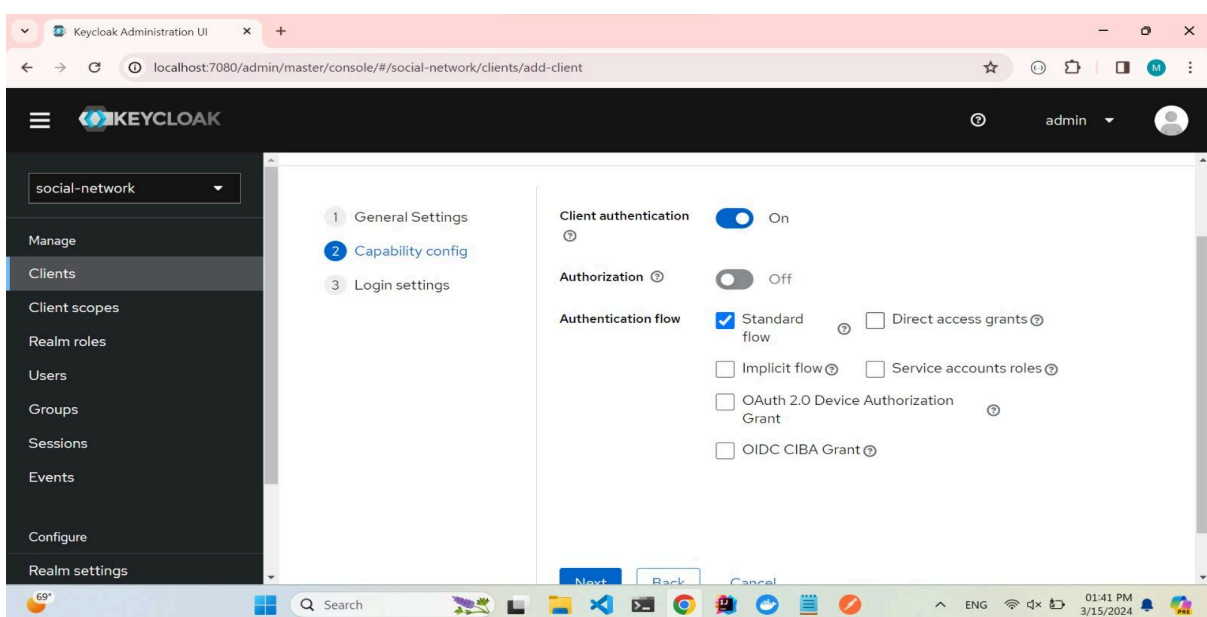
- Ensuite, nous avons créé un client en accédant à la section "Clients" dans le menu de gauche et en cliquant sur "Create".
- Nous avons configuré le client en choisissant "openid-connect" comme protocole et en spécifiant un nom pour le client, par exemple "SocialNetworkClient".
Configuration des paramètres du client :
- Après avoir créé le client, nous avons noté le "Client ID" et le "Client Secret" générés automatiquement par Keycloak.
- Nous avons également configuré d'autres paramètres du client selon nos besoins, comme les URL de redirection autorisées, les rôles, etc.



The screenshot shows the 'Add Client' page in the Keycloak Admin Console. The left sidebar has a dropdown menu set to 'social-network' and a list of navigation items: Manage, Clients (selected), Client scopes, Realm roles, Users, Groups, Sessions, Events, Configure, and Realm settings. The main content area has a breadcrumb trail: 1 General Settings, 2 Capability config, 3 Login settings. The 'General Settings' tab is active, showing the following fields:

- Client type**: OpenID Connect (dropdown)
- Client ID**: resource (text input)
- Name**: resource (text input)
- Description**: (empty text area)
- Always display in UI**: Off (toggle switch)

At the bottom right, there are three buttons: 'Next' (blue), 'Back' (grey), and 'Cancel' (grey).



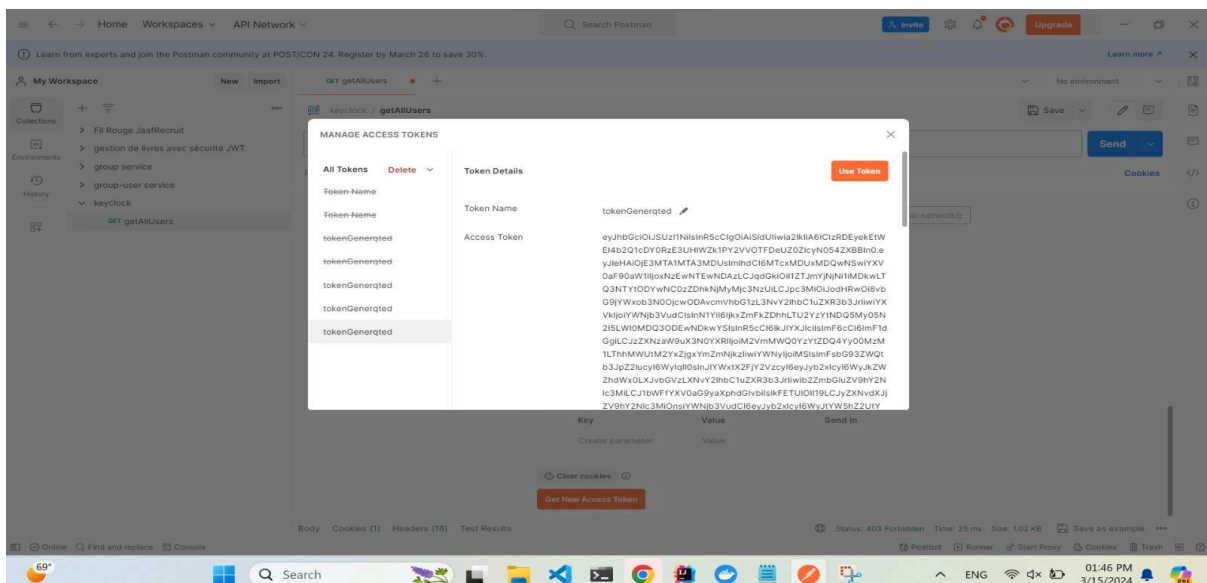
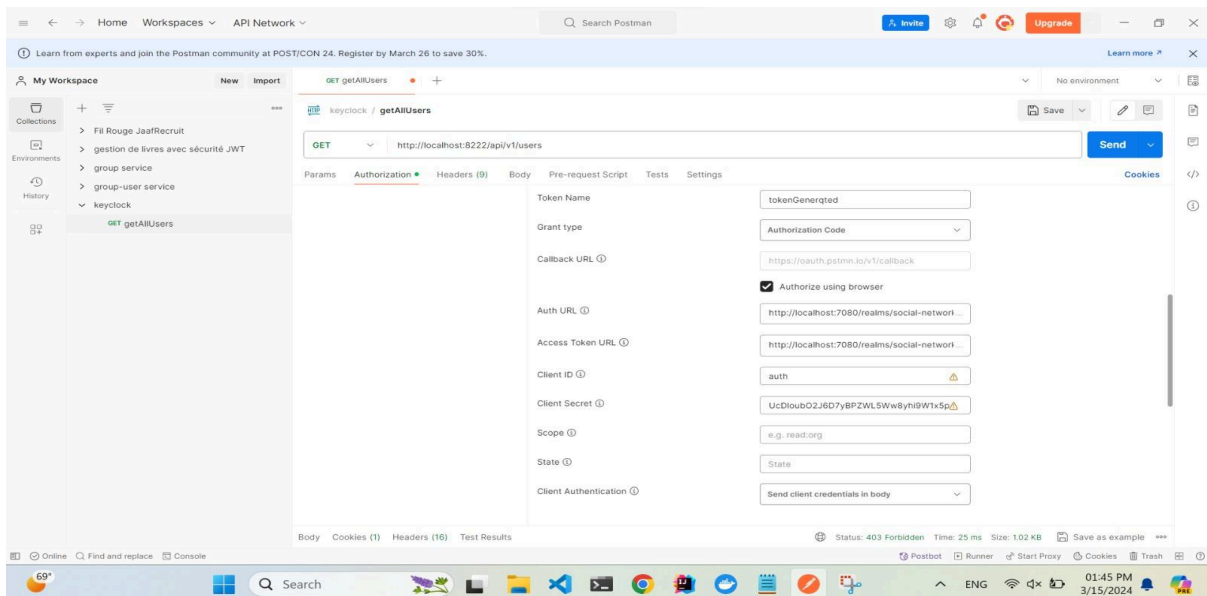
The screenshot shows the 'Add Client' page in the Keycloak Admin Console, with the 'Capability config' tab selected. The left sidebar is the same as the previous screenshot. The main content area has a breadcrumb trail: 1 General Settings, 2 Capability config (selected), 3 Login settings. The 'Capability config' tab is active, showing the following settings:

- Client authentication**: On (toggle switch)
- Authorization**: Off (toggle switch)
- Authentication flow**:
 - ☒ Standard flow
 - ☐ Direct access grants
 - ☐ Implicit flow
 - ☐ Service accounts roles
 - ☐ OAuth 2.0 Device Authorization Grant
 - ☐ OIDC CIBA Grant

At the bottom right, there are three buttons: 'Next' (blue), 'Back' (grey), and 'Cancel' (grey). The Windows taskbar is visible at the bottom of the screen, showing the time as 01:41 PM on 3/15/2024.

Test avec Postman :

- Pour tester l'intégration de Keycloak dans notre projet, nous avons utilisé Postman pour envoyer des requêtes à nos APIs sécurisées.
- Nous avons configuré une demande dans Postman en spécifiant l'URL de l'API à tester.
- Dans l'onglet "Authorization" de Postman, nous avons sélectionné "OAuth 2.0" comme type d'authentification et avons utilisé le "Client ID" et le "Client Secret" que nous avons obtenus précédemment.
- Après avoir obtenu un jeton d'accès valide de Keycloak, nous l'avons utilisé dans notre demande Postman pour tester l'authentification et l'accès à nos APIs sécurisées.



En suivant ces étapes, nous avons réussi à intégrer Keycloak dans notre projet Social Network, renforçant ainsi la sécurité et la gestion des identités au sein de notre architecture de microservices.

Conclusion :

L'intégration de Apache Kafka et Keycloak dans le projet social-network représente une étape cruciale dans le renforcement de la scalabilité, de la sécurité et de la réactivité de notre application de réseau social basée sur microservices. En adoptant ces technologies avancées, nous visons à offrir une expérience utilisateur fluide et sécurisée, tout en garantissant la croissance et l'évolutivité de notre plateforme. Connexia continue ainsi d'établir de nouvelles normes en matière d'architecture de microservices pour les applications sociales modernes.