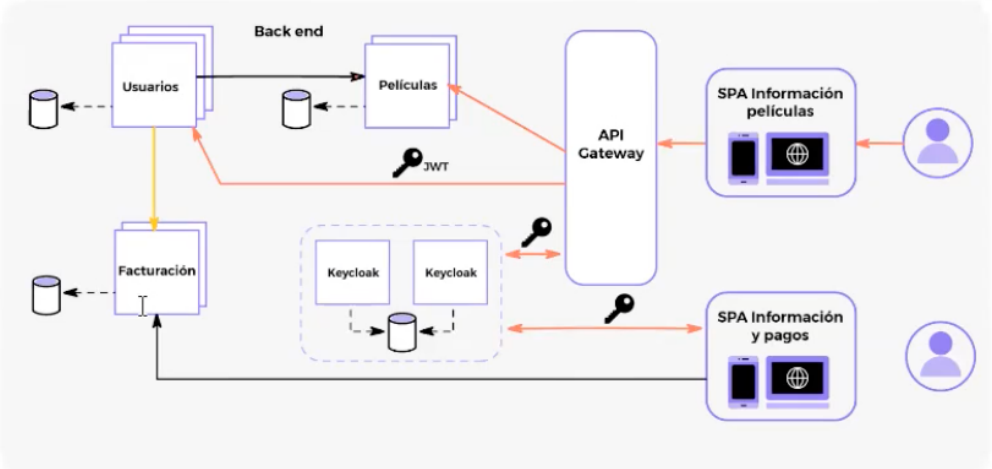
Parcial Especialización Backend II

Ecosistema de microservicios con Oauth2 y Keycloak como IAM

Alumna: Ana Laura Fidalgo

Camada: 1

Profesor: Rafael Rangel Sandoval



Resumen de instrucciones para correr el proyecto:

Ejecutar el archivo docker-compose.yml (Comando: docker-compose up)

Loguearse en la consola de administración de Keycloak. Crear el reino DigitalMedia importando las configuraciones con el archivo DigitalMedia.json.

Crear 3 usuarios (useradmin,userclient,userprovider) con los datos que figuran más adelante en este documento. Agregar cada uno a un grupo (admin, client, provider respectivamente).

Levantar los microservicios desde IntelliJ: 1.Eureka-Server, 2.Api-Gateway, 3.Peliculas-Service, Usuarios-Service, Facturacion-Service

Adjunto la colección de Postman para probar los endpoints. Posee variables de colección muchas de las cuales se setean automáticamente. Para comenzar a trabajar primero hacer request a los endoints 1 y 2 en la subcarpeta autenticación.

Consigna 1

Para correr Keycloak, abrir la terminal en la carpeta que posee el archivo “docker-compose.yml” y ejecutar el comando “docker-compose up”.

Keycloak se levantará en el puerto 8082:
http://localhost:8082

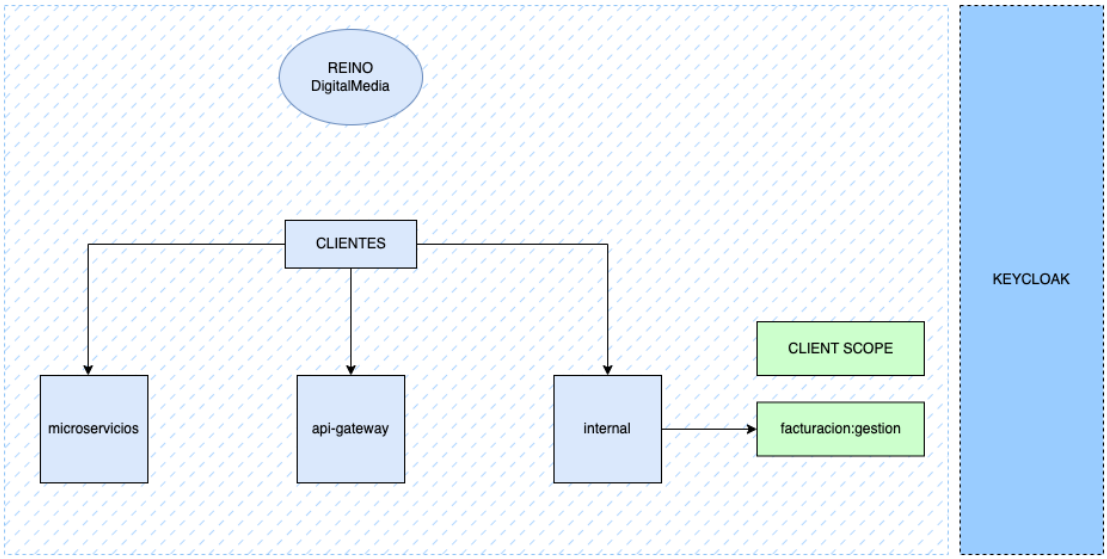
Ingresar al panel de administración con las credenciales establecidas en el archivo docker-compose:

- username:admin
- password: admin

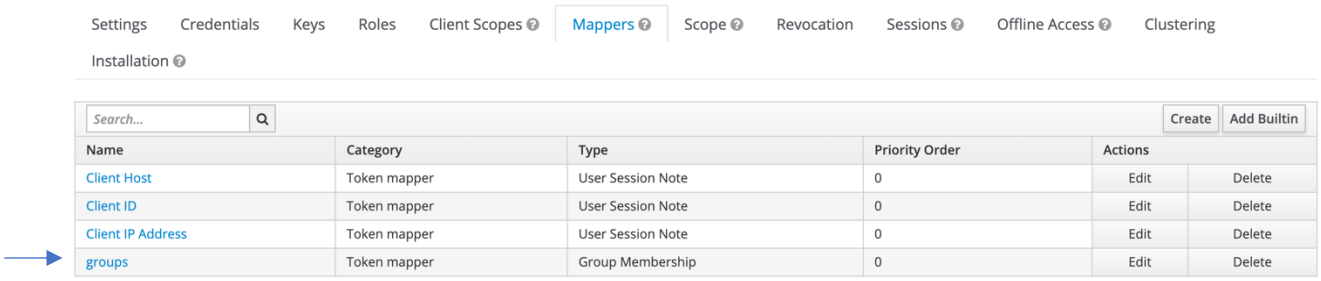
Consigna 2:

Una vez autenticado, crear un nuevo reino a partir del archivo DigitalMedia.json que está en la misma carpeta que el archivo “docker-compose”.

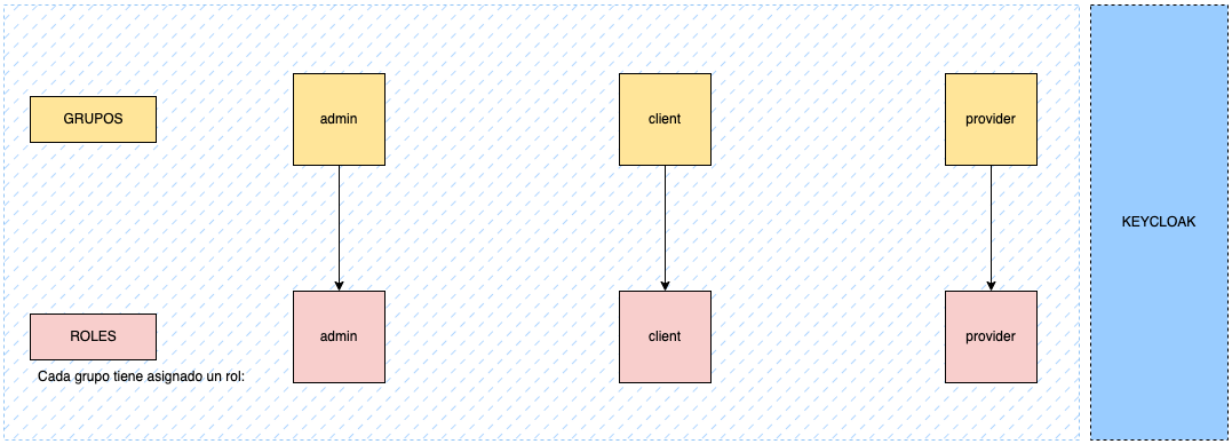
Para cumplir con las consignas del trabajo integrador, creé las siguientes configuraciones que se deberán importar automáticamente:



Cada uno de los tres clientes tiene creado un mapper de tipo Group Membership llamado: groups. Este mapper está agregado al Access Token para que el token contenga la información de los grupos y pueda extraerse para llevar adelante estrategias de autorización GBAC (group-based access control).



En el reino, he creado tres grupos: admin, client y provider. Cada uno de estos tres grupos debe tener un rol (admin, client y provider respectivamente).



Atención!

Para el correcto funcionamiento, el profesor deberá crear tres usuarios con las siguientes configuraciones (los usuarios no se guardan en el archivo .json al exportar/importar el reino de Keycloak):

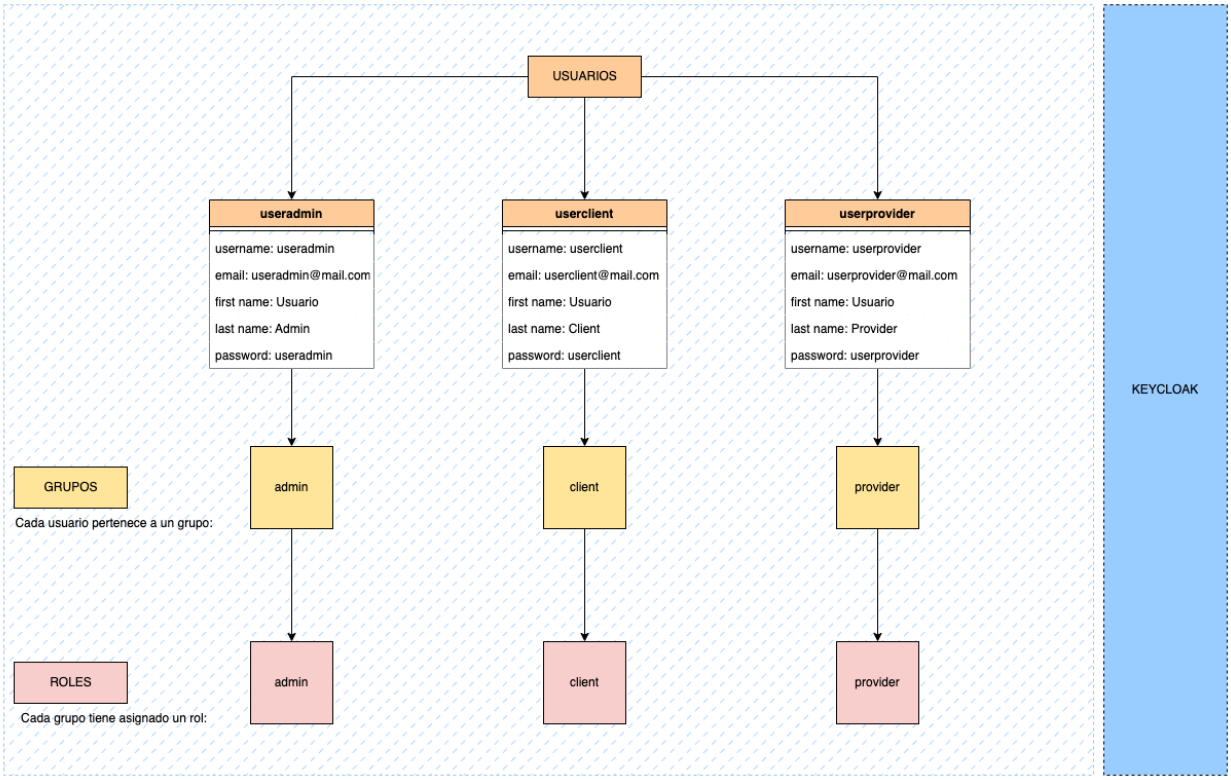
useradmin
username: useradmin
email: useradmin@mail.com
first name: Usuario
last name: Admin
password: useradmin

userclient
username: userclient
email: userclient@mail.com
first name: Usuario
last name: Client
password: userclient

userprovider
username: userprovider
email: userprovider@mail.com
first name: Usuario
last name: Provider
password: userprovider

Nota: no crear credenciales temporarias.

Luego, a cada uno de estos usuarios se les deberá asignar un grupo. El usuario admin se debe asignar al grupo admin, el usuario client se debe asignar al grupo client y el usuario provider se debe asignar al grupo provider. La configuración debe quedar de la siguiente manera:



Hasta este punto ya se han cumplido con las consignas 1 y 2 del trabajo integrador.

Consigna 3:

Una vez creados los usuarios, se puede proceder a levantar los microservicios.
El orden para levantarlos es:

1. Eureka Server (Puerto 8761)
2. Gateway-Service (Puerto 8080)
3. Peliculas-Service (Puerto 8086), Factuacion-Service (Puerto 8088) y Usuarios-Service (Puerto 8087) - en cualquier orden.

Según lo planteado por la consigna número 3 del trabajo integrador, creé los tres microservicios (Peliculas-Service, Factuacion-Service y Usuarios-Service) y les configuré la seguridad para que estos servicios actúen como servidores de recursos y que todos sus endpoints puedan ser consumidos únicamente por usuarios autenticados.

Según lo conversado en clase, Peliculas-Service y Usuarios-Service utilizan el cliente de Keycloak microservicios, mientras que Factuacion-Service utiliza al cliente internal, ya que este servicio no será consumido a través del Api Gateway.

Consigna 4:

Según lo planteado por la consigna 4, creé un servicio Gateway-Service (Puerto 8080) para mapear las urls de los servicios Peliculas-Service y Usuarios-Service. El Gateway-Service utiliza el cliente de Keycloak api-gateway. Para consumir los recursos de cualquiera de los dos servicios mapeados, el usuario debe primero autenticarse, de esta forma, por ejemplo, si el usuario no está logueado y quiere acceder al endpoint http://localhost:8080/movies el gateway lo va a redirigir al login de Keycloak. Unicamente permitirá acceso a los recursos una vez que el usuario haya sido correctamente autenticado (y suponiendo que cumpla con los criterios de autorización para ese endpoint en particular). Incluí en la configuración del Gateway el filtro de TokenRelay y un circuit breaker por si los servicios no estuvieran disponibles.

Según lo conversado en clase, no se puede acceder a Facturacion-Service a través del Gateway.

Nota: No logré que el flujo de autenticación a través del Gateway me funcione en Postman, únicamente funciona en el navegador.

Continuando con la consigna 4, configuré el servicio Facturacion-Service para que utilice un cliente de Keycloak propio (internal). Este cliente establece el uso de consentimiento para acceder a información sensible del usuario utilizando un nuevo scope del IAM: facturacion:gestion.

En este servicio, solo usuarios pertenecientes al grupo “provider” podrán dar de alta nuevas facturas. Además, el token deberá contener el scope facturacion:gestion.

En el servicio Peliculas-Service, tanto usuarios clientes o administradores pueden acceder a ver todas las películas, a buscar una película por su imdbId y a postear un comentario. Las operaciones de administración de las películas (POST – PUT – DELETE) sólo son accesibles para usuarios administradores.

Consigna 5

Al microservicio Usuarios-Service le agregué la dependencia necesaria para poder gestionar, mediante servicios REST, los usuarios de Keycloak.

El enpoint que obtiene una lista de todos los usuarios no administradores de Keycloak, valida que el usuario este logueado y que sea administrador.
El endopoint que crea un nuevo usuario de Keycloak, valida que el usuario este logueado y que sea administrador. Este método no le asigna un password, debe asignarse manualmente desde la consola de administración de Keycloak.
El endoint que busca un usuario de Keycloak por su ID, valida que el usuario este logueado y que sea administrador.

Consigna 6

Al servicio de Facturación-Service ya le había configurado la seguridad para que todos sus endpoints puedan ser consumidos únicamente por usuarios autenticados. Hasta este punto solo

usuarios pertenecientes al grupo “provider” podrán dar de alta nuevas facturas. Además, para poder dar de alta nuevas facturas, el token deberá contener el scope facturacion:gestion.

Para cumplir con la consigna 6, incorporé a este servicio la dependencia de Spring Cloud en pom.xml para utilizar Feign, configuré Feign con OAuth2, establecí el interceptor de request para inyectar el token de seguridad en todas las llamadas realizadas por Feign y definí la clase OAuthClientCredentialsFeignManager.

Luego creé el endpoint solicitado para que los clientes puedan visualizar todas las facturas asociadas a un usuario de Keycloak y sus datos. Este endpoint únicamente puede ser consumido por usuarios CLIENT.

Para crear este endpoint en Facturacion-Service, primero creé un nuevo endopoint en Usuarios-Service para buscar a un usuario de Keycloak con sus datos por nombre de usuario. Este endpoint también puede ser consumido únicamente por usuarios CLIENT.

Factuacion-Service se comunica con Usuarios-Service y consume este endpoint de Usuarios-Service a través de Feign.