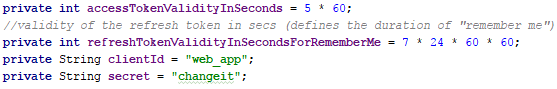
Este proyecto fue creado con JHipster el cual nos ayuda a crear el cascaron de un microservicio en back-end con Java Spring con una autenticación OAuth2.

En el proyecto se encuentran varios archivos de configuración con el OAuth. Los principales son tres: UaaConfiguration.java, UaaProperties.java y UaaWebSecurityConfiguration.java

UaaProperties.java:

En esta clase se encuentran algunas configuraciones en los nombres como son los tiempos en que el token estará activo así como el client\_id y el client\_secret de los cuales estos parámetros se pueden modificar al antojo del usuarios



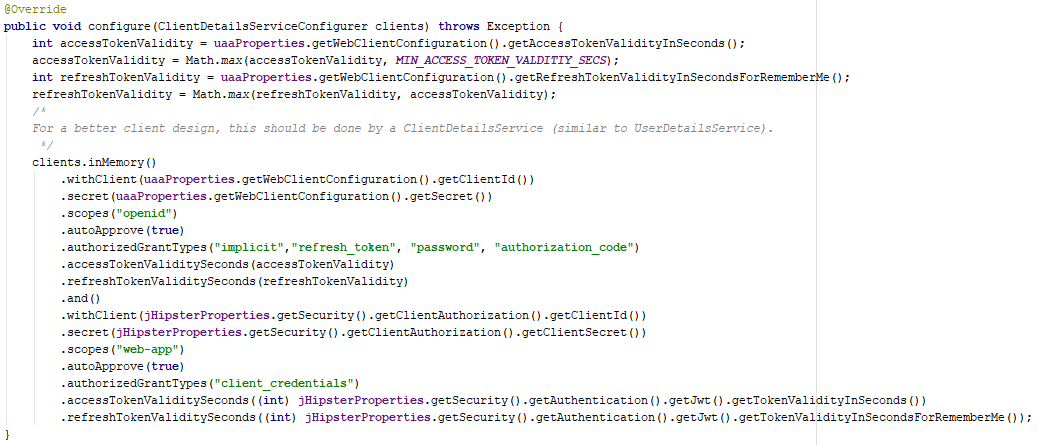
El primer entero se usa para colocar el número de segundos en el cual el token expirará y tendrá que volver a iniciar sesión

UaaWebSecurityConfiguration.java:

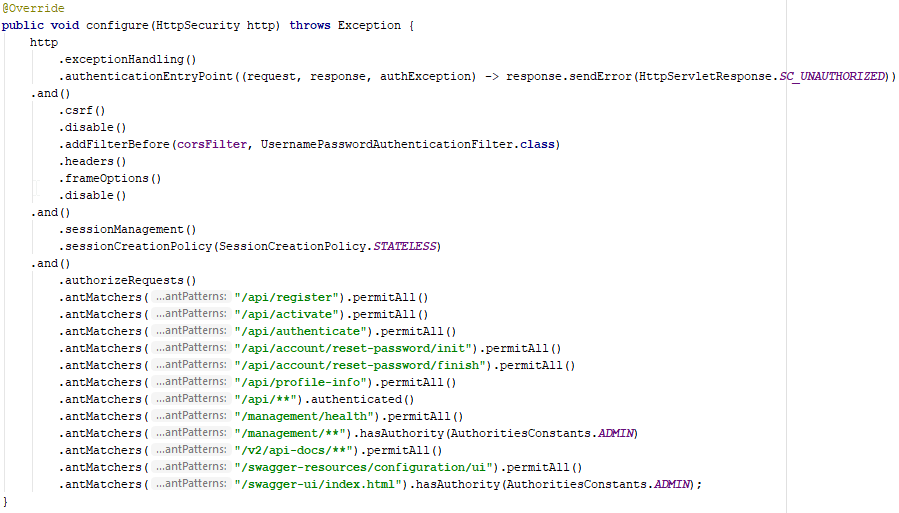
En este archivo no hace falta hacer ningún cambio ya que se quiere utilizar el mismo servicio de usuario y el manejador de autenticación que JHipster nos crea por default.

UaaConfiguration.java

Este archivo es el más importante de los tres ya que aquí se encuentra las URIs de las cuales estarán aseguradas por oauth, además de las configuraciones del client\_id, client\_secret, el tiempo de vida del token y el contenedor del token.



Este es el método en el cual se decide la configuración del oauth, en el cual se describe que tipos de gran autorización.



En este método se encuentra las URIs de las cuales podemos clasificarlas a los usuarios que pueden entrar a estar URIs dependiendo de sus roles, por lo que existen diferentes métodos para esta parte:



Este método permite a cualquier persona entrar al endpoint.



Este método permite entrar aquellos que tienen el Rol de ADMIN en los roles de cada usuario



En este último método, pueden entrar aquellas personas que estén autenticadas sin importar que rol sea. En el caso de JHipster la ruta de donde todas las demás caen es en /api/…. Por lo que de esta forma se considera que codas aquellas uris que se encuentran en esta principal serán accedidas por personas que este autenticadas.

Estos tres archivos ser encuentran dentro de la carpeta java/[nombre\_del\_proyecto]/config. Dependiendo del nombre que se le dé al proyecto es donde se encuentran estos archivos.

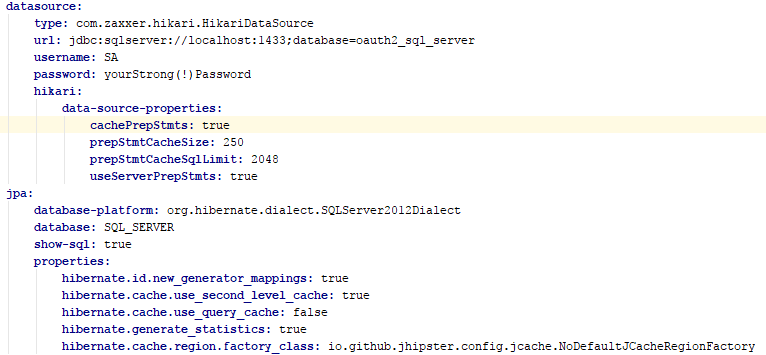
Otros archivos que son de importancia para JHipster son las propiedades del proyecto, en estos se encuentran el tipo de seguridad, la base de datos, los valores de las credenciales, entro otras, por lo que estos se encuentran en application.yml, application-dev.yml, application-prod.yml.

En estas propiedades las que más ocuparemos son los de las conexiones a la base de datos así como el dialecto para el JPA que se usa:



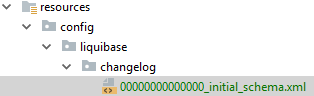
Este es el ejemplo que se uso para el oauth y se usa una base de datos en mysql el cual por el ejemplo usamos el usuario root con contraseña root para acceder a ella. Esta parte existe en los archivos application-dev.yml y application-prod.yml.

En cuanto a las propiedades con el sql server será de la siguiente forma:



El cual se debe de cambiar tanto la url de la base de datos como las credenciales para su acceso.

En cuanto a las tablas JHipster usa liquibase para el manejo de la base de datos (para tener mayor información pueden ver tanto la documentación de liquibase y <http://www.jhipster.tech/development/#using-a-database>)



Esta es la ruta en donde se encuentra el xml donde se arma para la creación de las tablas que JHipster usa

Un ejemplo de la tabla de usuarios que crea Jhipster es:



Al momento de correr y compilar el proyecto, liquibase se encarga de crear las tablas que se estén agregando al proyecto. De acuerdo a la documentación existen tres formas de crear las tablas:

1.- Con el sub-generador de liquibase

2.- Modificando a mano de los xml de las tablas.

3.- Modificando manualmente las tablas.

Ahora mostrare el ejemplo que se creó con la base de datos de MySQL, junto con Postman. En mi caso estoy usando IntelliJ como IDE para apoyarme:

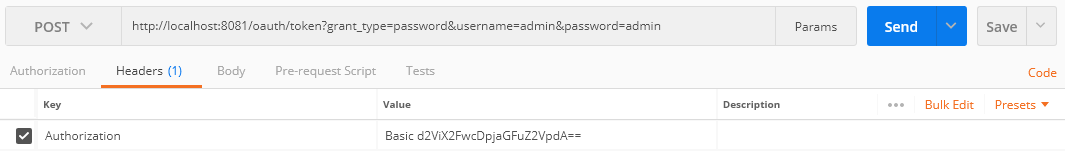
Cuando le demos click al botón de play  y termine de hacer las preparaciones necesarias del framework veremos una pantalla como esta:



En caso de no contar con el IDE tendremos que abrir una terminal y ubicarnos en la raíz del proyecto e introducir el comando: .\mvnw (en caso de Windows) y al finalizar mostrará lo siguiente:

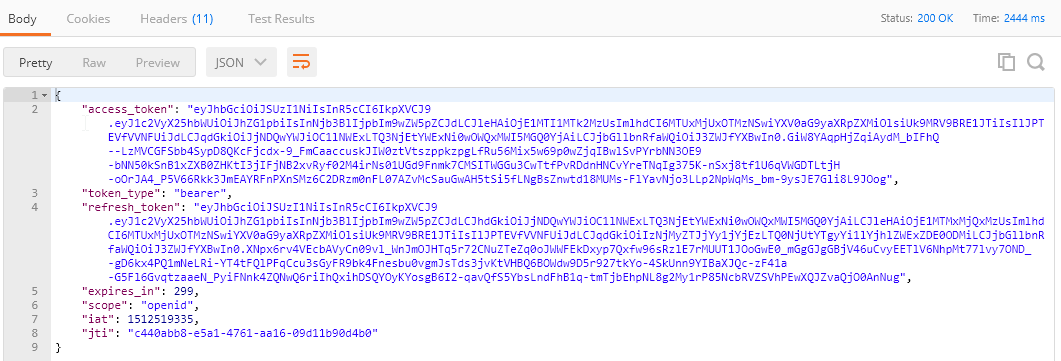


En estos momentos dejaremos corriendo el proyecto en la terminal y entraremos en el Postman para pedir el Token:

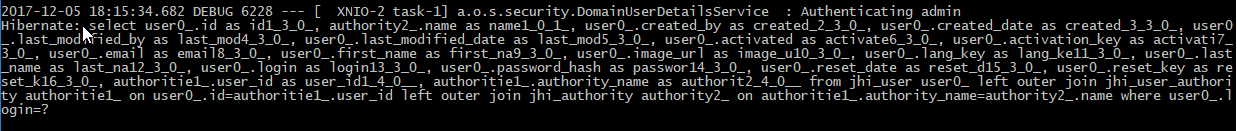


Para hacer la petición

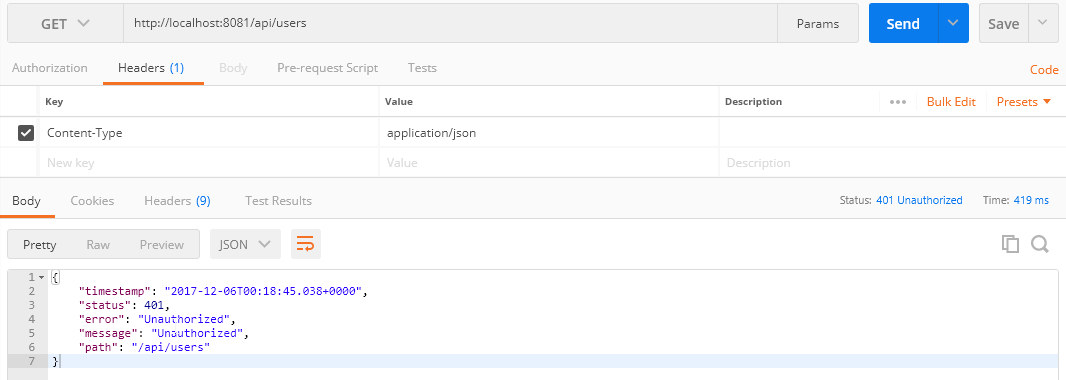
Y nos dará el resultado:



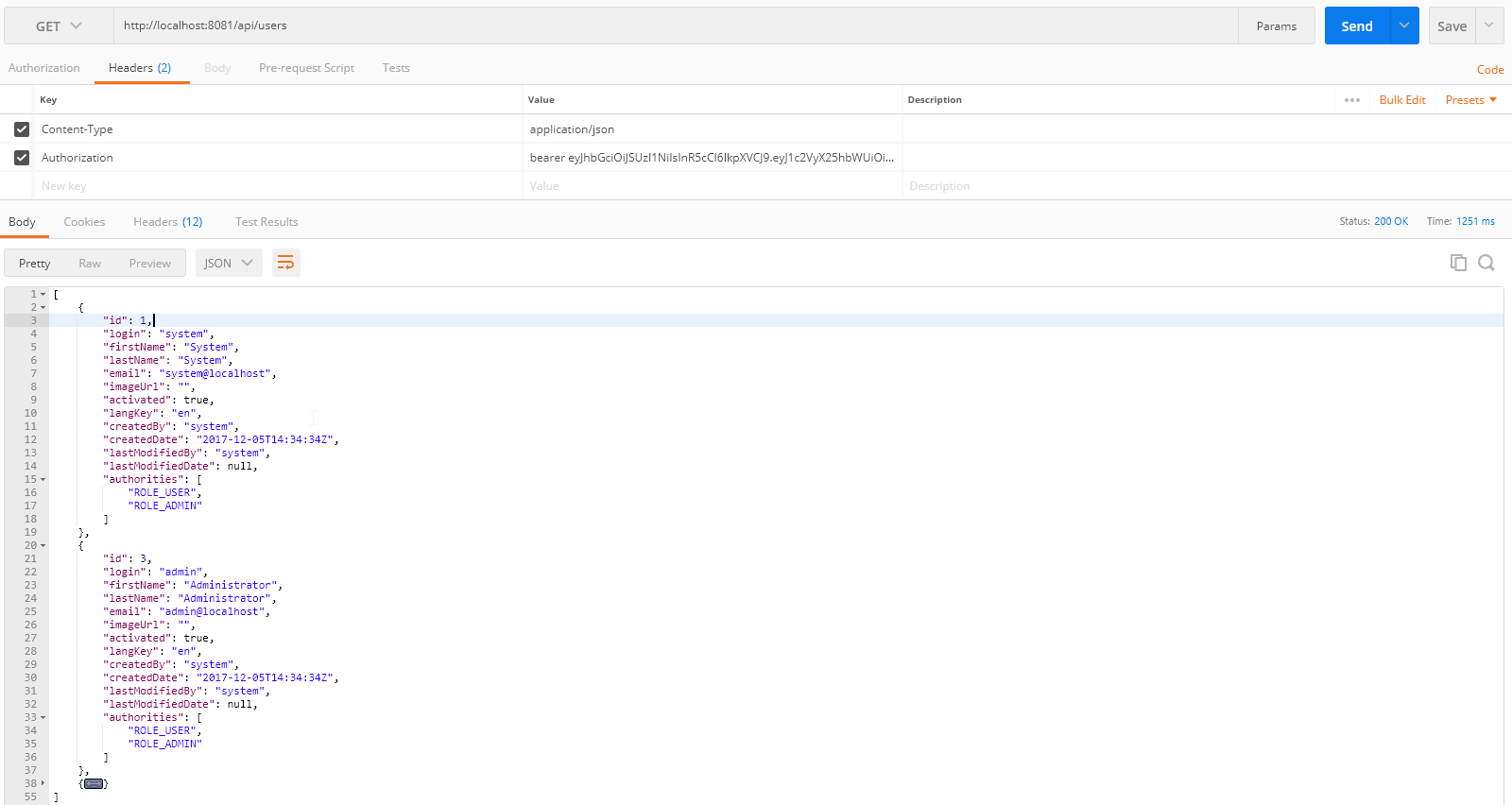
Mientras que en el servidor podemos ver que se realizó la petición:



Y si hacemos la petición sin tener la autorización no lanzara el siguiente error:



Pero si le colocamos la autorización, podremos ver el siguiente resultado:



Estos usuarios son creados por default por parte de Jhipster los cuales son:

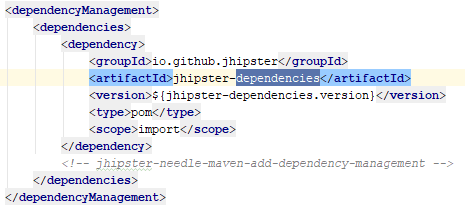
User: admin

Password: admin

User: user

Password: user

Por ultimo están las dependencias que se usan en el Proyecto:



Para el gestor de dependencias se está usando el de JHipster para la gestión de dependencias.

Algunas de las dependencias que está haciendo uso y que estamos explicando:

