Formato de Intercambio de Datos

El intercambio es realizado mediante peticiones a endpoints y JSON en el Request Body y en el Response Body.

Los JSON a usar en el request:

User:

```
{
"email": string,
"password": string,
"name": string,
"lastname": "string
}
```

RoleUpdater:

```
{
        "email": string,
        "password": string,
        "roles": String[]
}
```

Los JSON posibles como respuesta son:

CustomError:

```
{
  "code": int,
  "description": string
}
```

Authenticate:

```
{
“authenticate”: bool
}
```

Descripción URL de los Servicios

Todos los servicios tienen como url base {host}/v1

Endpoints de errores:

/customerror/list

Método: GET

Request Body: null

Descripción: Lista los errores con su código y descripción

Endpoints de Usuarios:

/user/add

Método: POST

Request Body: JSON de tipo User

Descripción: Se pasará el correo electrónico del usuario a crear, contraseña, nombre y apellido.El correo debe ser único en el sistema,el sistema genera un usuario. En caso de existir el usuario se retornará el error con código 101.

/user/role

Método: PUT

Request Body: JSON de tipo RoleUpdater

Descripción: Dado un identificador de usuario (mail) y contraseña, se puede agregar una lista de roles al usuario.El rol es un string, como por ejemplo Rol1, Rol2, etc. En caso del usuario no existir se retorna el error con código 102, en caso de la contraseña no coincidir se enviará el error 104. En caso de ya tener asociado uno de los roles que se pasan por parámetros no se genera un error.

/role/delete

Método: DELETE
Request Body: JSON de tipo RoleUpdater
Descripción: Dado un identificador de usuario (mail) y contraseña, se puede eliminar un conjunto de roles pasando como parámetros una lista de roles del usuario. En caso del usuario no existir se retorna el error con código 102, en caso de no coincidir la contraseña se enviará un error con código 104. En caso de ya no tener asociado uno de los roles que se pasan por parámetros se genera un error. con código 103 y en la descripción del error se debe indicar el nombre del rol que no se encuentra asignado al usuario.

/auth

Método: POST
Request Body: JSON de tipo User (solamente especifcando email y password)
Descripción: Dado un identificador de usuario y una contraseña, devolver un JSON con un campo true o false según corresponda.

Instalación

Se puede descargar el código fuente del repositorio https://github.com/rjmartinezuy/NoSQL_2021 Para conectarse al proyecto de Firebase personal, se debe de conseguir una key de firebase (Ver Guia: https://thepro.io/post/firebase-authentication-for-spring-boot-rest-api-5V) y después se debe de guardar la clave en la carpeta **Resources.**
Actualizar en src/main/java/com/nosql/comnosql/firebase/FireBaseInitializer.java la URL a la base de datos en **setDatabaseUrl()** y el nombre con el que se haya guardado la clave en **getResourceAsStream().**
Generar el archivo jar mediante el comando maven y ejecutarlo.

Configuración

En el archivo src/main/resources/application.properties se configura el puerto que va a utilizar la aplicación

Lenguajes

JAVA con Framework SpringBoot
Maven

Bases de Datos Utilizadas:

Firebase

De acuerdo a la realidad planteada en el Proyecto, el tipo de base de datos NoSQL que debiera utilizarse sería de tipo clave-valor. Al mismo tiempo, debido a la baja complejidad consideramos que también podría utilizarse una base de datos que

maneje documentos. Por ello, y debido a que no requiere instalación y se integra fácilmente con el modelo del Proyecto, es que optamos por utilizar Cloud Firestore de Firebase. Adicionalmente la cantidad de documentación y ejemplos facilitan su uso

Opcionales:

- ***Docker***:

  Teniendo instalado docker, y habiendo ya creado el archivo .jar, desde la carpeta raíz del proyecto:

  1) Crear la imagen con docker build -t [nombre de la imagen]
  2) docker-compose up -d
  3) docker run -ti [nombre de la imagen]

Nota: En el archivo dockercompose.yml se utiliza el nombre spring-nosql como nombre de la imagen, si no se va a realizar con el mismo nombre, debe de actualizarse en el archivo.

- **JMeter**:

  Teniendo una instancia de la aplicación corriendo, puede ejecutarse un JMeter para testing. Hay algunos ejemplos creados en este repositorio. Guia: https://jmeter.apache.org/

  Nota: Dentro del folder JMeter se encuentra el plan de pruebas preparado para los casos solicitados.