Assignment – Authentication API

Build a RESTful API which authenticates a customer based on his username and password. The API should expose the following endpoints:

1. one endpoint for creating an online banking account. Input data:
   - **the customer's account number** (a customer can create an online account only if his bank account exists in the **Accounts API**)
   - **a unique username**
   - **a password** (the password should have at least 6 characters)
2. one endpoint for authentication. Input data:
   - **the username**
   - **the password**

This endpoint returns a JWT access token in case of success (the token should be placed in the response body).

API constraints

1. if a customer introduces a wrong password for 3 consecutive times, then his account will be blocked for 24 hours.

Deliverable

1. The full implementation of the **Authentication API** using:
   - **Spring Boot 2.x**
   - **Jersey**, **Spring MVC** or **Spring WebFlux**
   - The project must be build with **maven**
2. The **Authentication API** must be exposed via TLS (8084) using **mutual authentication**
3. Unit tests
4. A swagger file which describes the **Authentication API** endpoints
5. *A bash script that starts the application* (**optional**)
6. *A bash script using CURL to call the endpoints* (**optional**)

Additional Info

For **data persistence**, you must use an **in-memory map**. The security of the data must be assured (keeping the password safe, for example).

To start the **Accounts API**, all you have to do is to execute the `./start-wiremock.sh` script. This API exposes a **mutual TLS connector** on port **8444**.

You must connect to the **Accounts API** from your application via TLS (you need this API to get the customer's ID based on his bank account)

A call to this URL: **https://localhost:8444/accounts/77853449** will return:

```
{
"iban": "NL24INGB7785344909",
"ownerId": "c3629d83-95f7-4966-9b67-76b13fe2cd5a"
}
```

In the **conf** folder, you can find the server **keystore** and the server **truststore**, which are required to expose your API on TLS.

The password for **all jks** files and for **all private keys** is **changeit**

The format of the jks is **JCEKS**. This information is useful for KeyStore initialization.

Advices

You can use **client.jks** to create the SSL context needed to connect to your API from Java.

If you want to use **CURL** to call your API, you need to first extract the private and the public key from client.jks so that you can specify them in the CURL parameters. To do this, you can use the `./key_extract.sh` bash script.

Please keep the solution as simple as possible

Don't add additional features! (It will definitely not be appreciated because it proves that you didn't understand the requirements.)

Don't create more classes that is necessary!