Finologee
Digital Finance

Development Assignment

```
public static void main(final String ... args) {
   // Hi & welcome! Good to have you here!
   // This is a homework task for assessing
   // your software development skills.
   // Please complete it for the next round
   // of discussions.
}
```

Application Project

Here is a small project that we propose for your implementation :)

What will be assessed:

- Architecture of the project
- Code quality and use of coding best practices
- Problem solving skills
- Delivery capabilities
- Instructions on how to run your application

Please read the full description before you start coding.

Try to complete the "Main requirements" within the given constraints. If you feel comfortable with some of the "Optional requirements", don't hesitate to complete them as well.

Requirements

In summary, you are asked to create a banking REST Web Service.

Main requirements

Authentication

Provide a rest endpoint for the authentication mechanism that validates the "username" and "password" of the client (provide a couple of users).

List the bank accounts

Provide a rest endpoint to list the bank accounts of the authenticated user (at least 2 bank accounts per users).

Create single payment

Provide a rest endpoint to create a single payment.

- The giver account should belong to the authenticated user.
- Payments to the same account number should not be allowed.
- Payments that exceed the available balance of the account should not be valid.
- Payments to this given list of forbidden accounts should not be valid (LU280019400644750000, LU120010001234567891).
- Giver account balance should be decreased when a payment is executed and receiver account balance – increased (If it belongs to the bank).
- Call an external web service to validate the IBAN format of the beneficiary (you can find a free service on the internet – REST or SOAP).

List all payments

For the authenticated users,

- Provide a rest endpoint to list all created payments ordered by creation date.
- Provide a rest endpoint to list all created payments to a given beneficiaryAccountNumber and within a given period ordered by creation date.

Delete payment

Provide a rest endpoint to delete a payment given its "id".

- The payment should belong to the authenticated user.

Update User Info

Provide a rest endpoint for the authenticated user, in order to allow him to update user information

- User should be able to update only the following fields
  - password
  - address

Logout endpoint

If your application relies on a "token" (session or other), provide an endpoint for the user to logout of your application. Meaning, provide a way to "invalidate" the "token".

Optional requirements

If you found this project too easy, feel free to give a try to these optional requirements. You don't necessarily need to consider them in the given order.

1. Payment execution process should be part of a single transaction.
2. Implement a pagination mechanism for the listing of created payments.
3. Register a fraud tentative when the user attempts a payment to forbidden accounts and block the account in case of more than 5 fraud tentative.
4. Provide a docker build(s) for your application so we can run it in containers.
5. Provide a swagger specification of your REST web service.

Constraints

- Use Java language (no scala, kotlin, or other)
- Use a Java Enterprise grade framework (Spring (Boot) and/or Java EE, or other)
- Use an H2/PostgreSQL database to store your entities (Please provide scripts to initialize database if not handled automatically by the application)
- Provide simple and clear explanations on how to run your application; The easiest the better.
- Provide a summary of which part of the requirements you have done and which part of the requirements you have not done.
- Potentially, provide the parts of optional requirements you have done.

Model

Here is a description of the minimum model to be used.
Feel free to change the structure and add properties if needed (in case of changes, justification would be appreciated).
Feel free to choose the type of the properties when not given.

BankAccount

- id
- accountNumber (iban format)
- List<User> users
- accountName
- List<Balance> balances
- status (enabled or blocked)

Balance

- id
- amount
- currency
- type (end_of_day or available)

Payment

- id
- amount
- currency
- BankAccount giverAccount
- beneficiaryAccountNumber (iban format)
- beneficiaryName
- communication
- creationDate
- status (can only be "executed")

User

- id
- username
- password
- address

Submitting the results

Once done with this task - please reply to our email with the summary of what is done. Send us the java-project and any other supporting artifacts (attached as a zip file or a link to the repo in the public source control service of your choice (GitHub / GitLab / Bitbucket etc.)

Good Luck!