Ebberød bank

Flow 1 project

Project information

Project name:
Ebberød bank

Flow & semester name:
Flow 1, Semester 2

School name:
CPHBusiness Lyngby

Group name:
Group B1

Group members:
Andreas Buch, Albert Emil Hyrde Jensen,
Albert Havn Kops & Aleksander Christian Rolander langkjær.

GitHub project:
Link

Requirements

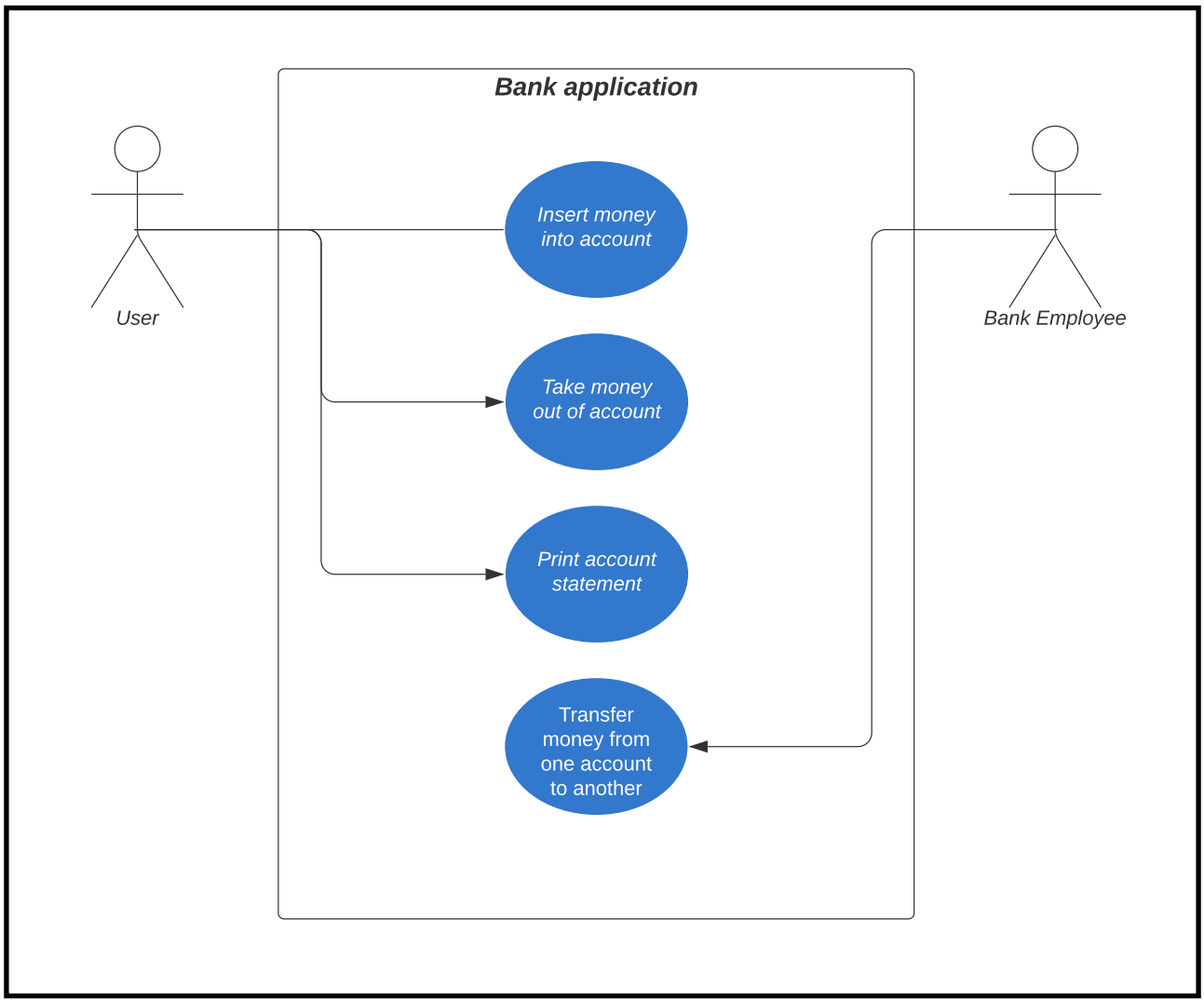
Functional requirements
- Program must have multiple customers.
- Each customer has one account associated with them.
- Customers must be able to put money into their accounts, take money out of their accounts, and they're not allowed to overdraw.
- Customers must be able to print out a transaction statement.
- Bank employees must be able to move money from one customer account to another, once again without overdrawing.

Non-functional requirements
- It's okay to work in whole numbers, as the program is only a proof of concept.
- The program must have a GUI but it can run in the terminal.
- Program must use data which persists in a mySQL database.
- Project has to be written in Java and be created as a Maven project.
- Use GitHubs "project & issues" feature for project oversight.

Project status

- En kort status over hvor langt I er nået (hvilke user stories har I implementeret).

Use case diagram



Use case diagram final, can be found in project folder under “PDF & UML”

User stories & acceptance criteria

User story:
As a customer of the bank, I want to be able to deposit money into my account as I accumulate more cash.

Acceptance criteria:
- Deposit amount needs to be added onto the account balance.
- Success message has to be implemented so customers know that the deposit has come through successfully.

User story:
As a customer of the bank, I want to be able to withdraw money from my account as I need it.

Acceptance criteria:
- Bank must check account balance against withdrawal amount before moving money, so that customer doesn't overdraw.
- Program must have exceptions in place for any errors, such as not having a high enough balance.
- Success message has to be implemented so customers know that a withdrawal request has come through.

User story:
As an employee of the bank, I want to be able to move funds from one customer account to another, so as to better help with our customers requests.

Acceptance criteria:
- Bank must check account balance before moving money, so that customers account doesn't go into the negatives.
- Program must have exceptions in place for any errors that might occur, such as input mistakes or not having a high enough account balance to go through with a given transfer.
- Success message has to be implemented so customers know that a transfer has come through.

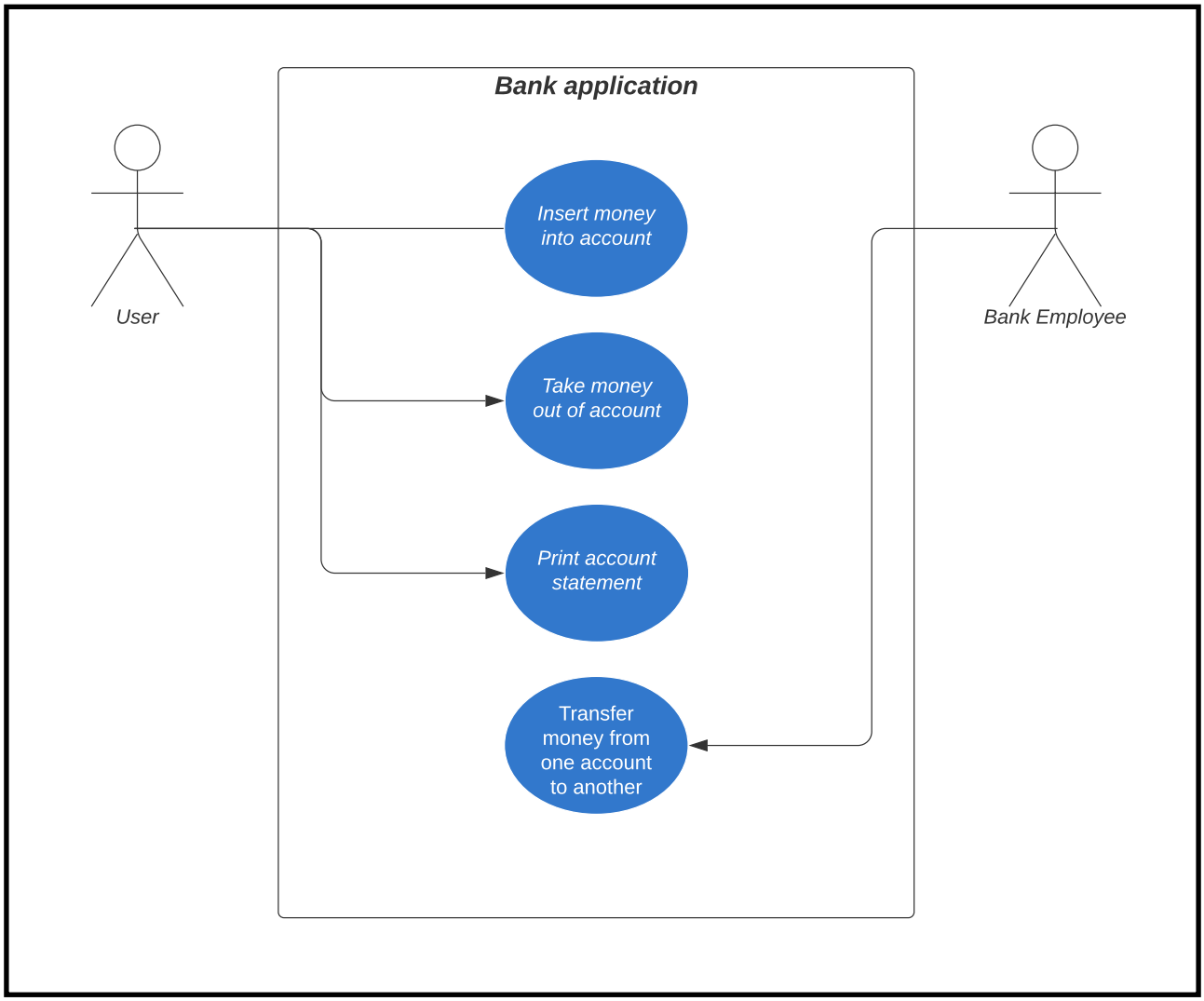
User story:
As a customer of the bank, I want to print out an account statement so that I can stay on top of how much I've spent in a given time period.

Acceptance criteria:
- Bank must be able to show current account balance to give a clear picture of a customers current financial state.
- Bank must be able to show an account statement which encompasses all three types of transaction types made on the account.
- Bank must also distinguish between these types, so that it's clear to the customer wether an amount was deposited, withdrawn or transferred to or from another account.

Given: (er ansat),
When: (vil se liste) af (Kunder),
Then: sort (KundeListe) efter (Navn), vise (KundeListe),

Given: (er ansat),
When: (vil se Liste) af (Konto),
Then: sort (KontoListe) efter (beløb størrelse), vise (KontoListe),

Enhanced entity relational diagram



Temporary placeholder for EER diagram

Reflection

- **En kort refleksion over hvordan det er gået med projektet.**
- **Kunne I finde ud af at bruge Git?**

Git bist kein problem meine freund.

- **Kunne I finde ud af at arbejde sammen?**

Teamwork was great from the beginning of starting work on the project. We might have begun a little late, but when we did, everyone in the group took it upon themselves to pick something to start working on. At this point it's still early in the project, and so we're not too organised yet. For that reason we decided to all work on different parts of the project, so that we could start building out the skeletal structure of the project.

Andreas started working on the pdf and use case diagram, while the two Alberts started working on the java project itself, building out the classes and program logic we'd need. Aleksander began working on the user stories and accept criteria.

- **Var der noget I eventuelt vil kunne gøre anderledes næste gang?**

Start the same day we get the assignment?