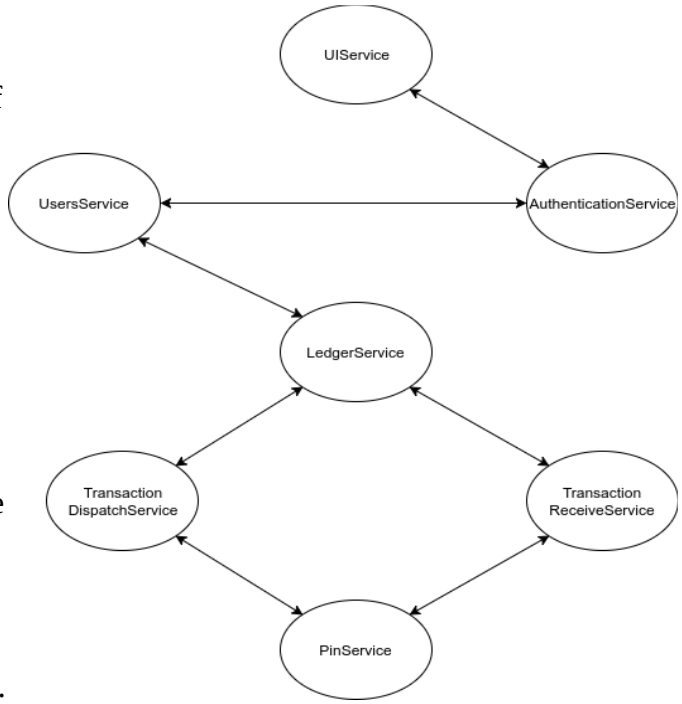
Our implementation

To implement this system we built a microservice-based system that incorperates 7 microservices(in the first version) which each can be ran seperately and have their own database. We took this approach with the intention of making the system agile, each service can run completely independently, possibly on different machines. We made use of a library that supports asynchronous requests to make the system very robust and scalable. Besides that we wanted to make as little use of 3rd party software as possible and have tried to limit the amount of libraries we use as much as possible.



The system contains the folowing services:
* LedgerService - keeps track of the balance of all accounts in the system, and updates the balance of accounts after transactions.
* PinService - Handles all incoming PIN/ATM requests and makes sure they are processed correctly in the system, issues Pin Cards to customers and keeps track of the accountNumber and customer each Pin Card belongs to.
* TransactionRecieveService - Handles all received transaction requests destined for an account in our system and makes sure these are processed correctly.
* TransactionDispatchService - Processes transaction request that are sent from an account in our system.
* UsersService - Handles all customer data(e.g. name, address, phone number) and links a customer to one or more account numbers.
* UIService - Handles all user input. Used to make most of the requests, for example a customer can request for a transaction to be executed, or request a new pin card.
* AuthenticationService - Authenticates a customer, keeps track of login information for customer accounts and issues cookies to customers when they log in.

Libraries

We made use of the Qbit library for asynchronous requests, Gson library for converting json strings to objects, Junit for testing, and mysql-connector-java for communicating with the mysql database. All these libraries are managed by gradle.

Assumptions/decisions

For everything that exists in the system there need to be requests to add/remove these things in the system.

An ATM transaction is just a transaction to a fixed account number, currently it will take any accountNumber as long as you specify that it is an ATM request. The same goes for deposits, we assumed that the ATM machine itself was out of scope of the system.

Closing a bank account will remove the account from the system.

Identifying to the system is done using a cookie which is set upon login.

The protocol is currently slightly different from the requirements in what exact reply is sent back, this will be updated in the first extension.