# Assignment 2

## Specifications:

Within our website, we decided to implement the following specifications:

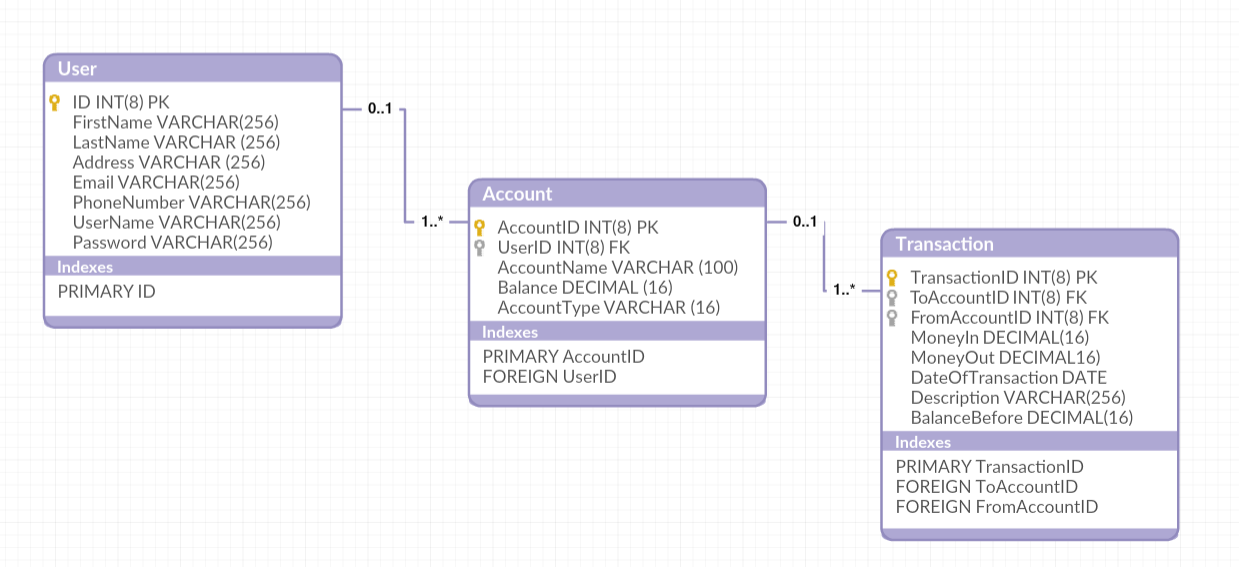
* Login/Logout functionality
* Functionality for users to create a user account
* Users can both open and close bank accounts
* A list of all of the user’s current bank accounts as well as corresponding balances and other necessary information is available on the homepage.
* A complete list of transactions involving each account can be viewed presenting the relevant information e.g. Date of transaction, what accounts were involved and so on.
* Users have the ability to perform transfers between their own individual accounts as well as payments from any of their accounts to any of the other users accounts.

## Design Choices:

We decided to implement these specifications as we deemed them the most crucial for a functioning bank application.

Concerning our structure, we’re using a Model View Controller Architecture such that all aspects of our design are grouped accordingly allowing for multiple pages to be designed simultaneously. We used an object oriented design, which allowed us to closely relate objects such as users (amongst others) to their database representations allowing conversion between the two seamless as well as clean code design. We used a relational database to store our necessary information such that we were able to have persistent information on our website (multiple users logging in and out as well as their corresponding information). Sessions were used for ease of querying allowing navigation to be optimally sped up, as well as dynamic phtml pages to be used that gave further instructions to users as to what is preventing them from continuing amongst other helpful hints. The home controller was used as a navigator controller directing pages accordingly based on user action whilst other controllers held functionality for just their corresponding models.

## Database Schema and Relations:



## Installation Instructions:

Simply unzip the zip file, navigate to the appropriate directory and docker-compose up. Navigate to localhost:8000 where you will be prompted to log in. From here, you can either login or create a user account. A dummy account has been set up for you that has a few accounts with money already in them as well as a few transactions already made for you to view.

Username: AGilman

Password: AGilman1

Whilst you can still set up your own user account and create bank accounts for that user. There would be no way for a you to insert money into your accounts (as a normal user). To get money in said accounts one would have to be able to access and edit the database or have other users send them money hence why an account has been set up for you to play with.

## Instructions for end user:

Navigation of our site is relatively simple and straight forward. Before being logged in, the user will have access to two pages: the login page; and the user create page. These each possess a feature from our specifications. Once logged in, there is a navigation bar along the top as well as a navigation sidebar that between them will navigate you to pages that allow you to perform almost all the remaining specified tasks. The only exception being the transactions. To view the transactions, simply click on the Account ID of the account whose transactions you wish to view and you will be redirected to a page where it will present just that. This information is available on the site however for ease of use is repeated here.

When filling out forms, pay attention to the hints beside each field. If your input is incorrect, you will be notified either by a small pop-up that will prevent form submission or a message appearing on form reload which will tell you what went wrong and how to avoid it happening again.