

Money Hub

Modest Software Engineering Project (MSEP)

Software Management Plan

Created by:

Sam Dressler & John Neis

CSCI 463- Software Engineering

2/24/2020

Table of Contents

Revision Summary	2
1. Introduction.....	3
1.1. Project Summary.....	3
1.1.1. Project Development Strategy.....	3
1.2. Product Description.....	3
1.2.1. Product Purpose.....	3
1.2.2. Product Scope.....	4
2. Project Planning.....	4
2.1. Schedule.....	4
2.1.1. Milestones and Deliverables.....	4
2.2. Estimation.....	5
2.2.1. Effort Estimation.....	5
2.2.2. Cost Estimation.....	7
Appendix.....	8

Revision Summary

Revision	Name	Description of Change	Date
1.0	Sam Dressler John Neis	Initial report design and layout, Cost and Effort Estimation	2/20/2020
1.1	Sam Dressler	Introduction, Project Description, Product Description, Schedule, Milestones	2/23/2020
1.2	John Neis Sam Dressler	Updated early cost estimation amount and added descriptions on the estimation factors. Updated product description	2/24/2020

1 Introduction

1.1 PROJECT SUMMARY

The development of Money Hub will be developed over a timeframe beginning with January 21st and ending no later than May 14th, 2020. To complete this project, a team of two members was created for which the technical work will be equally split. An additional role of team leader will be assigned during the development process. The team member that takes this position will be responsible for the management and assigning of tasks as well as communication between members and with the project assigner henceforth known as the *customer*.

The Money Hub project, henceforth referred to as the *project*, has multiple process requirements that must be completed before the final delivery. The team must analyze constraints and specify requirements, as well as design, implement and, test a prototype before demonstrating the functionality of the final system. Sections in the report will provide more detail on each of these tasks.

1.1.1 Project Development Strategy

These steps will be developed incrementally following an Agile workflow. Development sprints will have an average length of 30 days. At the end of each sprint a deliverable will be presented. Finally, in order to assure the integrity of the system, the development process will follow the guidelines outlined by common software engineering principles.

1.2 PRODUCT DESCRIPTION

For the remainder of the report, the Money Hub system will be referred to as the *system* or *application*. The system being created during this project will fit the requirement of being an information system. Furthermore, this system will be a query-based information system that will be used for the tracking and monitoring of a user's financial information.

1.2.1 Product Purpose

The purpose of designing such a system is to ease the stress burden that most people face when they are trying to manage their finances. Our team believes that having an account that a user can open and then have access to nearly every

important piece of financial will give certainty in a world filled with uncertainties, hence the name *Money Hub*.

1.2.2 Product Scope

The scope of the project is to give a user a snapshot of their finances all in a single place. This means that the user will be able to see their net worth, checking & savings account balances, remaining loan debt, and many more useful pieces of information and analytics. For example, an account within the application will be able to add their account information for a checking account they have with Capital One, loan information they have with Direct Loans, and finally an investment portfolio opened in Robinhood. **

The benefits of a financial hub that includes all this information is the accurate representation of a user's net worth. Other applications that generate your net worth do not include the user's debts that may be in a separate account. Some limitations that will be highlighted in a later version of the report will be which accounts are eligible to be included in summary.

***Disclaimer - The company names used in this section are currently not affiliated with the Money Hub System and were just used as an example of what the scope of the project could include.*

2 Project Planning

Planning is the first crucial step in the development of the project. The customer expects us to develop and present a schedule as well as a budget. In addition to the development of these items, the customer also expects us to operate as close to the bounds of these as possible.

2.1 SCHEDULE

As stated in the project summary, the development of the project began mid-January and will continue through the beginning of May 2020. The development cycle for this prototype is quite short so the complexity of the prototype will need to be adjusted accordingly while still having the expected functionality.

2.1.2 Milestones and Deliverables

According to the team project specification in the syllabus for the course, there will be four deliveries as well as a final presentation that demonstrates a proof of concept for the project.

These deliveries are outlined in the following table:

Milestone	Deliverables	Date Started	Date Due
Delivery I	<ul style="list-style-type: none"> Cost & Effort Estimation Project & Product Specification Schedule 	2/18/2020	2/24/2020
Delivery II	<ul style="list-style-type: none"> Project Plan Documentation Software Requirements Specification 	2/18	3/24/2020
Delivery III	<ul style="list-style-type: none"> Software Design Documentation 	-	4/23/2020
Delivery IV	<ul style="list-style-type: none"> Software Test Documentation Project Source Code 	- 1/18/2020	5/14/2020
Demonstration	<ul style="list-style-type: none"> PowerPoint Presentation 	-	5/5/2020- 5/7/2020

**The deliverables past the first are subject to change due to possible changes to the syllabus as well as possible changes communicated during class.*

A visualization of the deliveries is laid out in the following table. The green section includes preliminary instruction for the project as well as the brainstorming period for the project. The gold sections are the areas where the team will be focused on assembling a deliverable option to the customer. The blue section represents the continual design, development, and testing that is associated with the Agile software development life cycle. Finally, the orange section indicates the deadline for the presentation.

January			February			March			April			May			
18th	24th	30th	12th	24th	29th	12th	24th	30th	12th	23rd	30th	12th	7th	14th	
Project Planning															
			Delivery I												
					Delivery II										
								Delivery III							
											Delivery IV				
			Continuous Software Design, Implementation, and Testing												
													Final Presentation		

In this equation, Effort is measured in person months, A is a constant of 2.94, B is the complexity factor ranging between 1.01 and 1.24, and M is a multiplier which considers reliability and complexity of the product (RCPX), reuse of the product (RUSE), difficulty of the platform (PDIF), experience of the personnel (PREX), the capability of the personnel (PERS), required schedule (SCED), and personnel support facilities (FCIL), and size is measured in thousands of lines of code.

$$M = RCPX * RUSE * PDIF * PREX * PERS * SCED * FCIL = 0.123$$

Factors for M:

$$RCPX = 0.85$$

$$RUSE = 0.75$$

$$PDIF = 0.5$$

$$PREX = 1.1$$

$$PERS = 0.7$$

$$SCED = 0.5$$

$$FCIL = 1$$

The values which contribute to the final value of M were chosen from a standardized table based on a rating from 1 to 5, where 1 is low priority and 5 is high priority. This does not necessarily mean the values shown for RUSE, PREX, and FCIL are low priority, it simply means the priority was used to determine the value of the factor.

$$B = \frac{(\sum Factors)}{100} + 1.01 = 1.17$$

Factors for B:

$$Precedentedness = 4$$

$$Development Flexibility = 5$$

$$Architecture and Risk Resolution = 2$$

$$Team Cohesion = 2$$

$$Process Maturity = 3$$

The value which contribute to the final value of B were ranked on a scale from 0 to 5 where 0 is considered extra high, and 5 extra low. This is counterintuitive, but this is the convention of the model.

The size of the system is estimated to be about 1,250 or 1.25 thousand lines of developer written code. Auto generated code is not considered in the effort estimation.

With these calculations completed, we can now estimate the time frame for the project.

$$Effort = 2.94 * 1.25^{1.17} * 0.123 = 0.47 PM$$

The result for this estimation is 0.47 PM which comes out to be around 83 hours.

This was calculated by taking the number of weeks in a month and multiplying it by 40. This will give the number of hours a full-time employee would spend working in a month. To tie in the estimate and the team's expectation, we first multiplied this by the result of the effort equation. Once we had that amount, the final step was to divide it by the number of team members working on the project, which in this case was 2.

The result showed that we expect to spend approximately 42 hours over the course of the project life cycle designing, implementing, and testing code, as well as writing, revising, and preparing deliverables for the customer.

2.2.2 Cost Estimation

The estimated time frame for completing the project, from beginning to completion of the system, will take about 0.47 months. Using time frame, the dollar amount of the developmental prototype of the system was calculated to be \$3,330.28. This amount comes from the number of hours our team expects to work multiplied by \$40/hour.

APPENDIX

Figures

Tables

Money Hub

Modest Software Engineering Project

Software Requirement Specification

Created by:

Sam Dressler & John Neis

CSCI 463- Software Engineering

4/1/2020

Table of Contents

1. REVISION HISTORY
2. INTRODUCTION
 - 2.1. Document Purpose
 - 2.2. Product Scope
 - 2.3. Intended Audience and Document Overview
3. PROJECT DESCRIPTION
 - 3.1. Product Perspective
 - 3.2. Product Functionality
 - 3.3. Users and Characteristics
 - 3.4. Operating Environments
 - 3.5. Design and Implementation Constraints
 - 3.6. User Documentation
 - 3.7. Assumptions and Dependencies
 - 3.7.1. *Assumptions*
 - 3.7.2. *Constraints*
 - 3.7.3. *Dependencies*
4. PROJECT REQUIREMENT SPECIFICATION
 - 4.1. Client User Interface Requirements
 - 4.2. Server Requirements
 - 4.3. Security Requirements
5. SYSTEM SPECIFICATION
 - 5.1. Data Modeling
 - 5.1.1. *Data Flow Diagram*
 - 5.2. Behavioral Modeling
 - 5.2.1. *State Chart*
 - 5.2.2. *Use Case Diagram*

1. Revision Summary

Revision	Name	Description of Change	Date
1.0	Sam Dressler	Initial report design and layout Introduction sections Report Description sections Project Requirement Specification	3/29/2020
	John Neis	Requirements Models -Use case, state chart, relationship diagram	3/29/2020
1.1	Sam Dressler	Added additional requirements and description for the product perspective model.	3/31/2020
	John Neis	Descriptions for models Server requirements Final touch ups to format	3/31/2020

2. Introduction

2.1 DOCUMENT PURPOSE

Money Hub, henceforth known as the *project*, is the product whose requirements will be specified in this document. The requirement specification shall cover the current functional and non-functional requirements as well as any design or process constraints.

The product is split into two main partitions, the Money Hub Client, henceforth known as the *client*, and the Money Hub Server, henceforth known as the *server*. The four specifications noted in the above paragraph will be specified for each of the partitions.

In addition to the specifications, this document will contain detailed models that will model the data and behavior of the system. These models will include a state chart and use case diagram. Additionally, an entity relationship diagram will be included to show a rudimentary connection between the user and the data in the system.

2.2 PRODUCT SCOPE

Money Hub was launched with the goal in mind of simplifying the user's financial situation. The goal of creating Money Hub was to give users access to a product that allows them to improve their financial competency.

Money Hub will provide the user with access to a variety of their accounts in one place. Having a mirror of their checking, and savings accounts, allow the user the benefit of being able to track their spending and saving.

Where Money Hub goes beyond your typical banking website, is its access to showing the user their investment portfolios being used in online investment firms. On top of that, one of our goals in creating the system is too be able to see what debts the user has in car, student, and other various loans.

Creating a comprehensive summary of this information in one place that the user can have easy access too will achieve the objective of increasing the financial intelligence of the population of the United States.

2.3 INTENDED AUDIENCE AND DOCUMENT OVERVIEW

The rest of this document contains a further description of the project and its environment as well as limiting factors such as constraints and dependencies. Additionally, this document will provide requirement specifications and certain models detailing the usage of the system.

The main purpose of this document is to present the customer (Dr. Hassan Reza Ph.D.), with the contents summarized in the previous paragraph. This document shall be considered a living document throughout the development phase as the requirements shall be influenced and modified based on feedback from the customer.

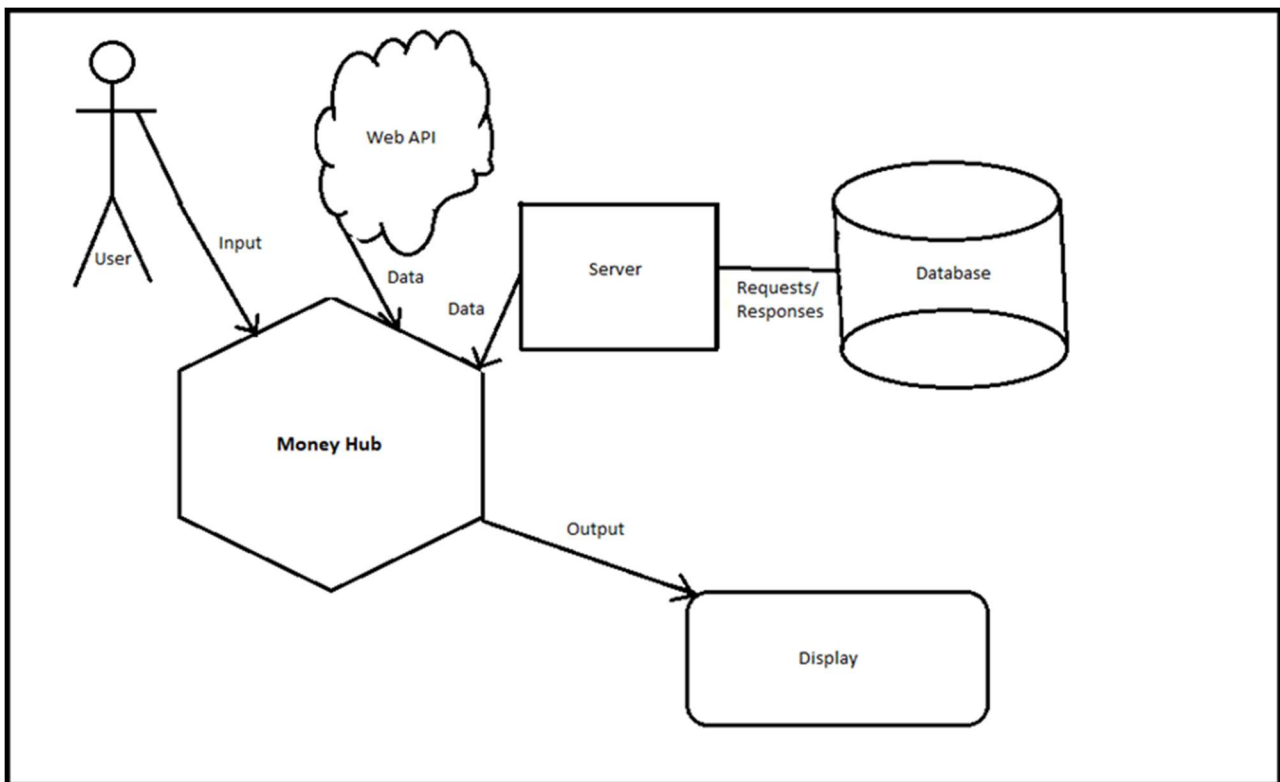
While the customer is the main motivator for writing this document, the contents can also be used by various other parties. The next important of these will be the people who are charged with the actual development of the system such as the software developers and testers. For those parties, the critical sections to read are found in section three. Readers in those categories can jump to that section to begin learning the specific specifications of the system as well as to begin studying the information and data flow in the models.

For readers in the categories of customer, project managers, marketing staff, or documentation writers, continue reading into section two. This section will give a broader description of the project that will help in further work that will help in marketing, planning risk, or writing insightful documentation. Lastly, the information from section two will make understanding the final section easier to comprehend.

3. Project Description

3.1 PRODUCT PERSPECTIVE

Money Hub is a product that was envisioned as an entirely new system. In the information age, nothing is more important than having the facts. This is especially true in our own lives. Opening an app and seeing all the important financial information is something that our team believes is crucial to survive in today's fast-moving world.



This model shows how the user will interact with the Money Hub system which is tied to external components. The web API and Server both pull information from sources that hold information for the user. Once the data is entered into the system, the processing will be executed and output to the user through the user interface on the display.

3.2 PRODUCT FUNCTIONALLITY

This system runs concurrently by connecting to a server through client applications. Additionally, there will be multiple web-based API's that allow our third-party software to extract account information from banks and lenders and feed it into the client.

First time users will create an account that is then stored on the server before being logged onto the Money Hub “Home” page. Here the user will see what the web APIs return to the client. This information will include the amounts for the accounts that the user has connected. At the same time, the information loaded from the web is then stored into that user’s account information on the server.

For users to have the balance of an account show up in the client, they will need to log into the respective company’s website through the client. Connecting to the accounts this way ensures that Money Hub never directly sees the user’s password for an account but still has access to the amounts they are trying to see inside of client application.

Returning users will log into the client and have their credentials validated through the server. Once into the client, the application will begin updating the balances shown in the application.

Users will be able to sync their Money Hub account with accounts from other firms by logging into the respected accounts through the client.

3.3 USERS AND CHARACTERISTICS

The usability of the system will be heavily considered when designing the user interface. The reason for this is so that whether the user is a recent college grad just gaining a real income for the first time, or a Millennial looking to get their finances in check, the experience will be one of ease.

Our team wanted the characteristics of the program to be one that is attractive and clean. Many times, applications are cluttered or difficult to follow from page to page. In designing a page like this, it will make sure that you don’t need to be a computer expert to use the system.

Reading the financial information will also be a breeze because all the amounts will be visible on a single page. Additionally, analytics performed on the account information will make it clear as to what the user’s financial situation is so that they can decide what their next correct step is.

3.4 OPERATING ENVIRONMENT

We live in a world where we are constantly on the move. In order to keep up with the pace of our lives, we need to develop a product that can be accessed on the go. To achieve this goal, The Money Hub application will eventually be developed to be used on the web, by browsers such as Google Chrome, Microsoft Edge, or Mozilla Firefox.

Currently the system will be a simple desktop application due to the short-handed development phase length and the lack of knowledge in our team in the area of web-based development. The limitations will be elaborated further on in clause 2.7.2. The operating system that the prototype will be able to operate on will be Windows 10.

3.5 DESIGN AND IMPLEMENTATION CONSTRAINTS

During the development process of a prototype for Money Hub, there are a few challenge areas that would need to be overcome in the development of a full fledged system.

1) **External Data Availability**

Access to external accounts takes time and permission. For the development of a prototype like this we have neither, so the data being fed into the system is non-existent. To develop the project fully, time and energy would need to be invested into getting firms such as banks and lenders to allow us access to their data through an API. Another constraint that comes from volatile data availability is the time it takes to access the data from the external firm.

While this process may seem daunting, there are already proofs of the concept out there. Many personal financial management tools are out there that implement the connections to firms that would be needed in this product.

2) **Data Security**

There are a few different levels of security that will be necessary. For example, the client will need to log in but where will the data used in validating the account be stored? In order to provide security for the user's information, that validating information will need to be stored in a database on a server rather than on the client.

3) **Portability**

Currently, Money Hub is a desktop application. Since we have a limited time frame to develop the prototype, we had to choose an operating system to develop on. For us we chose Windows 10, which means we won't have time to make sure that all the functionality works on another host running Linux or MacOS.

4) **Internet Access**

Because the client sync's new financial data from the web and validates its logins via a server, if a user is trying to log into the client without internet access, then they will not be able to access their account.

3.6 USER DOCUMENTATION

Utilization of the application and the service held within were designed so that the user would have minimal issues when learning to use the system. The user interface will be designed such that it is self-explanatory. However, one area where problems may arise is syncing an external account. To account for this, help will be available to the user via an internal help button near the option to add an account.

3.7 ASSUMPTIONS AND DEPENDENCIES

3.7.1 assumptions

During the planning for the development of this system, certain assumptions had to be made. These assumptions are those came during the preliminary brainstorming of the system and more may be added further into development.

- 1) The account information for all the user's that are added during the prototyping phase will be the same. These amounts will need to be hard coded into the prototype so that they can be loaded into the client once they log in.
- 2) The instance "web API" means all the API's that would be required to retrieve account information from banks, investment firms, and loan venders.
- 3) In the prototype, the web API will return the hard-coded values for a few accounts that will be decided during the design phase.
- 4) The application will be build using Java, C#, and an SQL database with the interface being designed using Visual Studio.

3.7.2 Constraints

The development process for the project will also face constraints on its development.

- 1) Time and effort to complete the prototype is limited. The entirety of the development process will take place over just a couple of months so complexity of the system will consider such constraints.
- 2) The system prototype will not be able to interface via an API with online firms. While this concept is possible, there must be a period of communication and development between organizations in order to ensure the security, legality, and feasibility of such ventures.
- 3) The money being displayed in the system for a given user will have not actually exist in the real world.

3.7.3 Dependencies

The development and implementation of the full Money Hub system will depend on the cooperation of partner firms in order to give the user access to the information that they desire. Additionally, once the system is fully operational, it will depend on these firms giving access to account information on a regular basis with low downtimes.

4. Project Specification Requirements

Requirement Specification Key

Token	Description
FR	Represents a functional requirement.
NFR	Represents a non-functional requirement.
<#ID>	three-digit number indicating the ID of the requirement.
<#ID>C	Indicates the requirement involves the client.
<#ID>S	Indicates the requirement involves the server component.
<#ID>SE	Indicates the requirement involves the security of the system.

4.1 CLIENT USER INTERFACE REQUIREMENTS

The user interface is one of if not the most crucial components of the system. This interface will contain multiple pages that allow for the user to access the remainder of the system and its functionality.

ID	Component	Description
FR001C	User Interface	The client shall validate credentials and log a user into the system in less than five seconds.
FR002C	User Interface	The client shall display the options to exit and minimize the program on every page, including the login page.
FR003C	User Interface	The client shall display the option to log out after a user has been logged in successfully.

FR004C	User Interface	The client shall correctly navigate to a new page when a button is clicked.
FR005C	Client-Server Communication	The client shall send a user account login ID and a password, ideally encrypted, to the server.
FR006C	Client-Server Communication	The client shall receive and display account information from the server.
NFR007C	User Interface	The client shall be able to connect to the internet and provide timely connection to the partner firms that the user has accounts with. Loading of this information and storing to the database shall not exceed 10 seconds.

4.2 SERVER REQUIREMENTS

The Server will be the means by which the client will communicate to access the database.
The server will be implemented using the Java programming language.

ID	Component	Description
FR001S	Database	The database shall record user account login IDs, password, a list of accounts and balances, and ages of the respective accounts.
FR002S	Client-Server Communication	The server shall query the database using login information received from clients in order to generate a login token.
FR003S	Server	The server shall sanitize any data received from the client, in order to avoid SQL injections.

4.3 SECURITY REQUIREMENTS

The client and the server will both have aspects that would be disastrous if they were not secure. Client's will trust the platform to ensure the safety of their information for their external accounts.

ID	Component	Description
FR001SE	Security	The client will not track account credentials to external firms.
FR002SE	Security	User login credential validation will be performed on the server side of the application.
NFR003SE	Security	Users of the Money Hub shall authenticate themselves using a username and password that meet the requirements for passwords.
FR004SE	Security	User's password will consist of a minimum of eight characters. The password shall contain a capital letter, a number, and a special character.

5. System Specification

5.1 DATA MODELING

5.1.1 – Entity Relationship Diagram

Shown below is a rudimentary model, demonstrating the relationships between relevant data that will be used. A user will consist of account number, password, first name, and last name attributes, with account number and password being candidate keys. An account will consist of account number, balance, and type attributes, with the account number being the primary key. Here, a single account can only be held by a single user, however a single user may hold many accounts.

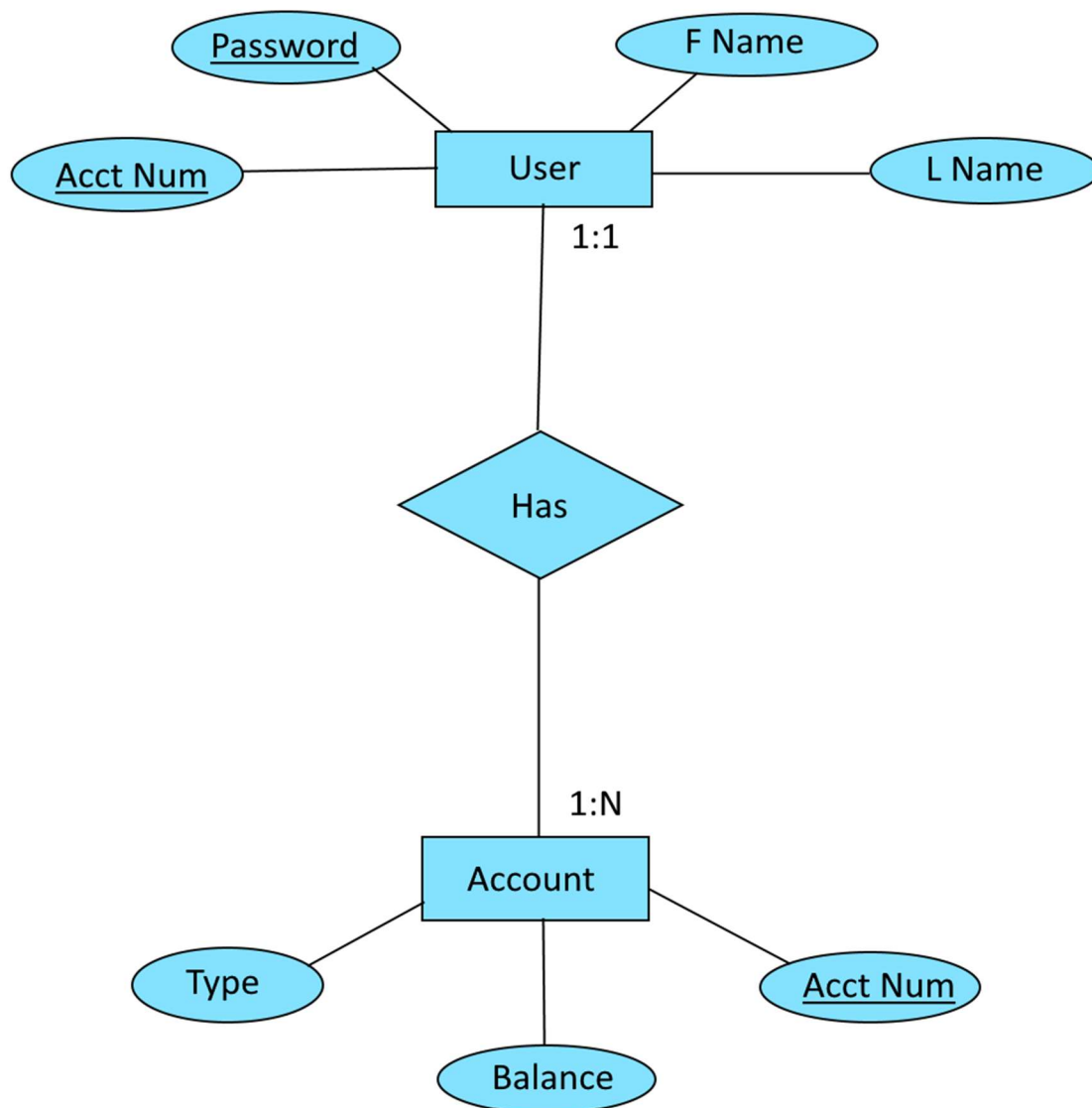


Fig. 4.1.1 – Shows how the data of relevant to the database is structured

5.2 BEHAVIORAL MODELING

5.2.1 State Chart

Shown is a state chart which is meant to describe the program flow. The system consists of three major components: the client, the server, and the database. Initially, the server will be in a standby mode as it listens for incoming requests from clients. The client will initially show a login screen. When the user logs in, the request must be verified by the server, which will query the database in order to determine whether or not the login information is good. At this point the server will send back a login token to the client. If the token is good, the client can display the information returned by the server. If the token is bad, and error message will be displayed, prompting the user to re-enter their login credentials.

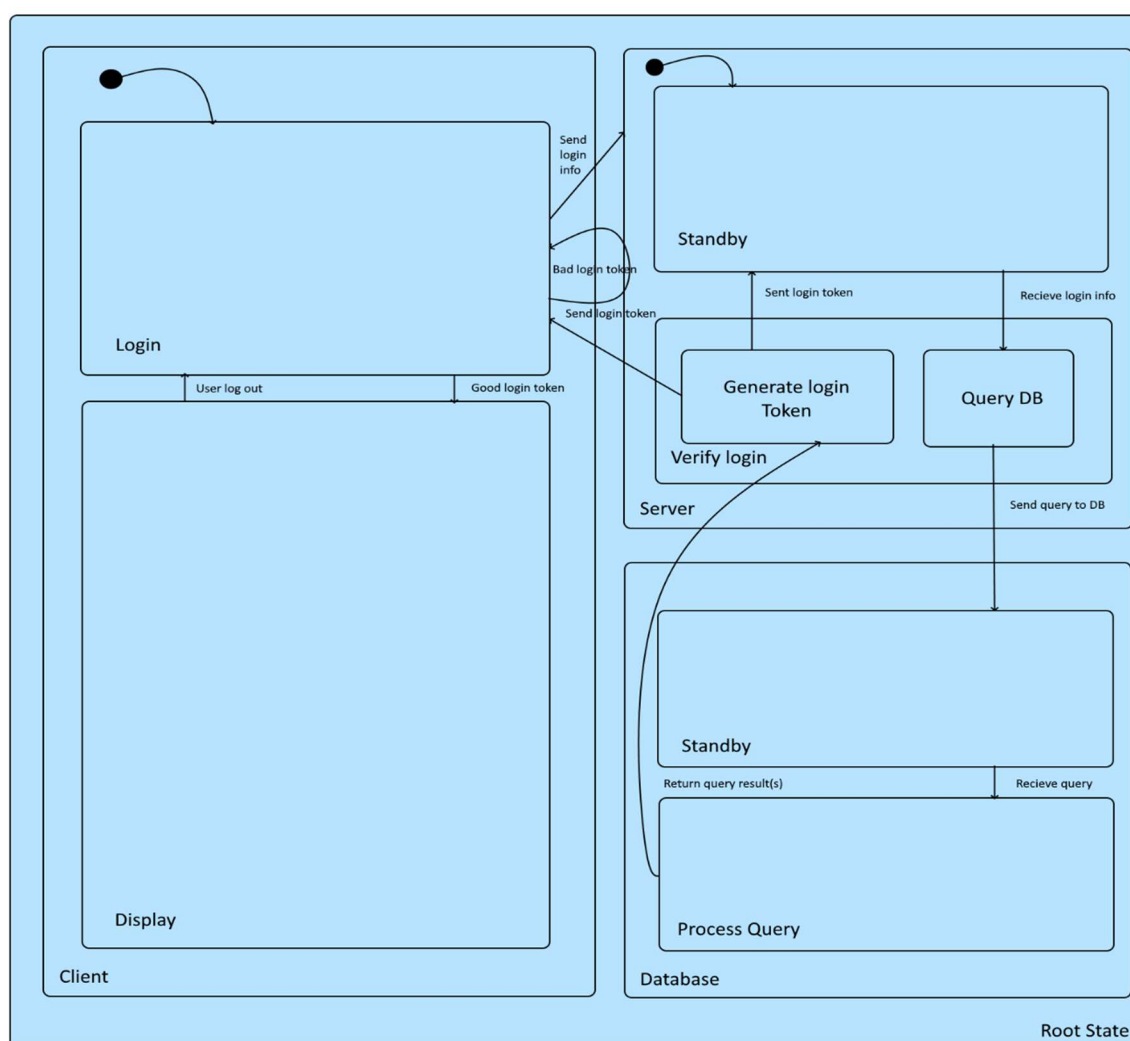


Fig 4.2.1 – The state chart diagram demonstrates the flow of control and data throughout the execution of the program. Here, the client and server are treated as separate states, as one is active while the other is inactive. This, however, can be extended to include multiple concurrent sessions between multiple clients and the server.

5.2.2 Use Case Diagram

Shown is a use case diagram, meant to illustrate the various cases and users that might utilize this system. Clients will be able to view their account information, freeze or unfreeze use of any payment cards tied to their accounts, wire money from one account to another, and monitor transactions they make on those accounts, for example.

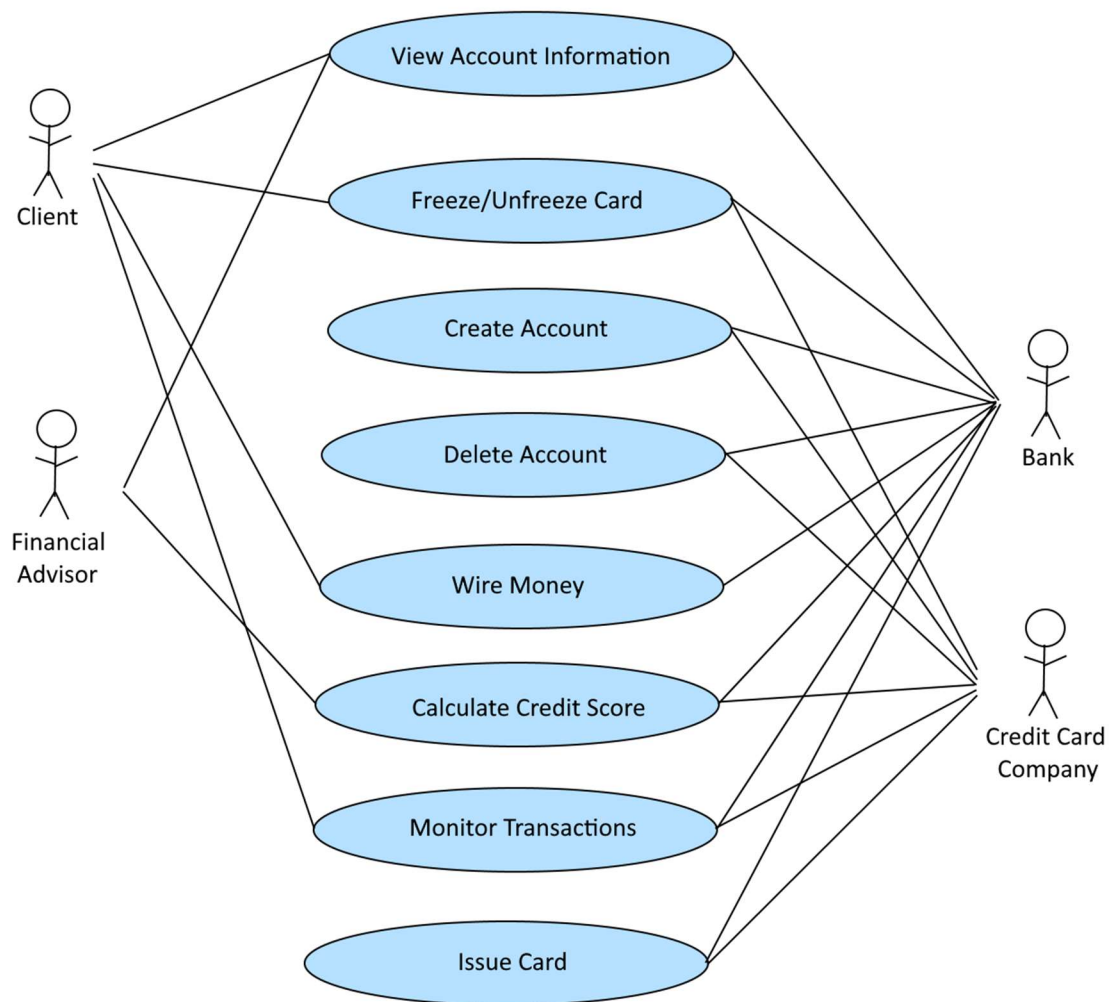


Fig 4.2.2 – The Use Case Diagram demonstrates the various uses the system will provide. Throughout the development process, more use cases may be discovered, so this may not be a totally comprehensive visual.

Money Hub

Software Engineering Project

Software Design Documentation

Created by:

John Neis & Sam Dressler

CSCI 463- Software Engineering

4/29/2020

Table of Contents

Revision History

1. Introduction

- 1.1. *Document Purpose*
- 1.2. *Product Scope*
- 1.3. *Intended Audience*

2. Design Considerations

- 2.1. *Assumptions and Dependencies*
- 2.2. *General Constraints*
- 2.3. *Goals and Guidelines*
- 2.4. *Development Methods*

3. Architectural Strategies

- 3.1. *Programming Language*
- 3.2. *Component Reuse*
- 3.3. *Further Work*
- 3.4. *User Interface*
- 3.5. *Software Interface*
- 3.6. *Error Detection and Recovery*

4. Software Architecture

- 4.1. *System Architecture*
- 4.2. *System Decomposition*

5. Policies and Tactics

- 5.1. *Coding Guidelines and Conventions*
- 5.2. *Plans for Testing Product*
- 5.3. *Maintaining Product*
- 5.4. *Organization of Source Code*
- 5.5. *Generating System Deliverables*

6. Detailed System Design

- 6.1. *Classification*
- 6.2. *Definition*
- 6.3. *Responsibilities*
- 6.4. *Constraints*
- 6.5. *Compositions*
- 6.6. *Uses and Interactions*
- 6.7. *Resources*
- 6.8. *Processing*
- 6.9. *Detailed Software Design and Interfaces*

Revision Summary

Version	Name	Description of Change	Date
1.0	Sam Dressler	-Document structure and initial layout -Initial complete draft of section 1. Introduction -Initial complete draft of section 3. Architectural Strategies -initial complete draft of section 5. Policies and Tactics	4/29/2020
2.0	Sam Dressler John Neis	-added assumption to the design considerations section 2.2 -client considerations for section 6 -Completed sections 2, 4, and server and database components of section 6.	4/30/2020

1. Introduction

1.1 DOCUMENT PURPOSE

This document contains the detailed design of the Money Hub system. The design will cover the user interface, the backend server, and the SQL database. Furthermore, the document will cover limitations that affect the development of the system. System architecture, policies, developmental strategies, and preliminary talk of testing will also be included. For a general overlook of the document, consult the table of contents found at the top of this report.

1.2 PRODUCT SCOPE

Money Hub was launched with the ideal in mind of simplifying how people manage their finances independently. The primary goal of creating Money Hub is to give users access to a simple and easy to use application that enables tracking of expenses and savings. The tools being designed into the product will prepare customers for financial challenges that will no doubt present themselves in life.

Money Hub will provide the user with visibility to a variety of their accounts in one place. Example of the accounts that can be viewed by the user can range from savings and checking to investments and loans.

Where Money Hub goes beyond your typical personal finance application is its access to showing the user their investment portfolios being used in online investment firms. On top of that, one of our goals in creating the system is too be able to see what debts the user has in car, student, and other various loans.

Money Hub will create a comprehensive summary of this information that is visible all in one place. The project team believes that allowing users to see all this is the first step in increasing the financial intelligence of the general American population.

1.3 INTENDED AUDIENCE

Software design documentation is primarily written with the purpose of allowing developers, system architects, and software testers a document to consult when they have questions regarding specific areas of the software.

In addition to developers, the design documentation will also serve the purpose of giving the customer a view of how the product is being developed. Referred to as the customer in the previous sentence, Dr. Hassan Reza, will be the primary evaluator of the system and this

document will contain the necessary information to understand how the application will be structured.

2. Design Considerations

2.1 ASSUMPTIONS AND DEPENDENCIES

At the current phase of development, the Money Hub system considers several assumptions that will affect the system. The first of these, is in the testing of the product. Because the product will rely on displaying information that is taken from external parties, the application will have to set these fields with pre decided information in order to display the desired results in the prototype.

Secondly, the system relies on the use of network communication to retrieve information, as well as communication between the server application and the database system. The communication between these components must be reliable to ensure unimpeded use of the system. As of the present time, the project will work on the Windows operating system. Support for other operating systems may, or may not, appear in the production release.

2.2 GENERAL CONSTRAINTS

Hardware used to run this product must have a semi-reliable internet connection. Without this, there can be no communication between the client and the server, and thus an end user will never be able to view any relevant information.

To that end, the server must always remain operational. If a user needs to view their account at unusual times, for instance in a time of emergency, the user must be able to access their information. If the server needs to be made inoperable (for servicing/maintenance for example), ample warning should be provided to all users of the system.

Network communication will utilize a reliable protocol (TCP) in order to ensure there is no data loss or corruption when displaying user account information.

For the server, the application must be able to interface with a relational database system. The relational database system will be used to store and maintain user data.

2.3 DEVELOPMENT METHODS

Development of the product was conducted via the waterfall method. Small, proof of concept module prototypes was developed alongside the documentation and design of the product following a model more akin to agile development.

3 Architectural Strategies

3.1 PROGRAMMING LANGUAGES

The Money Hub System Architecture is currently being developed as a desktop application. Production versions of the system will allow for downloads and installation off an external site. The prototype of the system being discussed in this report, will consist of three main components. Each of these components will be implemented using unique programming languages.

The first component is how the users will interact with the system. This part of the architecture will be discussed further in section 3.4 of the system design documentation. The language chosen to implement this will be C#. One of the main reasons C# was used was the ease in which it allows for graphical user interfaces to be developed through Microsoft visual studio. The language also offers the ability to connect to servers that are written in java.

The second component, the server, will act as a conduit for which the user will communicate to the database. Java has options that make it a prime candidate because it can communicate with clients written in other languages.

The final component of the architecture is the database. The Structured Query Language, otherwise known as SQL, was chosen in order to maintain the relative information for the system. Since Money Hub is a query-based system, SQL makes sense for the development team to use for this.

3.2 COMPONENT REUSE

The current design for the system does not make use of any existing external components. The system will however offer means for the system to reuse internal data structures and functions.

3.3 FURTHER WORK

The System architecture will eventually need to include means in which it can query information from partner banks and firms off the internet. This web API is not currently in development because of the cooperation needed from these third parties. Additionally, the system would eventually be developed for mobile and web use in order to increase the access of the system to clients.

3.4 USER INTERFACE

The Money Hub user interface is designed with ease of access of user information in mind. Since the client application is currently being designed for Windows, there is a lot of flexibility with how we organize and size the display. A careful balance of how much information is available on one page and application size is needed in order to make sure

that the user has an enjoyable experience while using the system. The interface will need to adhere to response time requirements outlined in the SRS in order to offer responsive and interactive experience while navigating the client.

3.5 SOFTWARE INTERFACE

Software interfaces are how the user client, accessed through the user interface, will communicate and query the database. Requests from the user will be coded and sent via a server connection that contain the request in json or xml format.

3.6 ERROR DETECTION AND RECOVERY

From the current point of development, many errors will not be discovered until product testing begins. Testing and error detection will be discussed further in future documents. The current design phase is being completed with initial proof of concept of interaction for the system components as its primary focus.

4 Software Architecture

4.1 SYSTEM ARCHITECTURE

The product will operate as a query-based system where the users will send requests to a server via the clients User interface. The centralized server will allow each of the unique client applications to request the information needed by the current user logged on that

port.

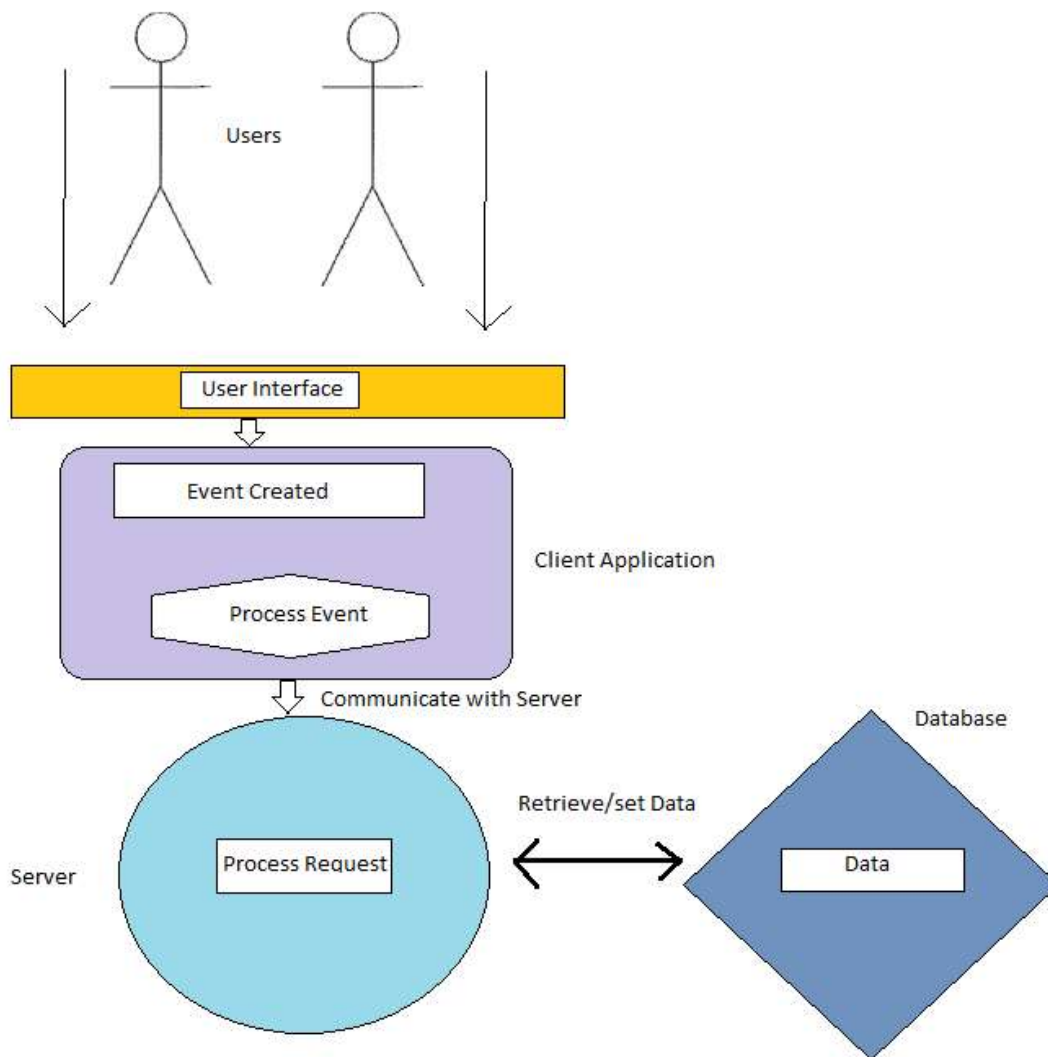


Figure 1 System Architecture Diagram

4.2 SYSTEM DECOMPOSITION

Client: The client module will request the user provide account login information. In the final release of the product, the client will encrypt the login information before sending it to the server. Once the server responds, the client will display either the information requested by the user, or a failed login message, prompting the user to resubmit their information.

Server: The server will remain in a state of standby, listening for login requests from clients. The server will parse the information received from the client, and if the login information is valid, will build a string of data attributed to the user and send it back to the client. In the final release, this information will also be encrypted.

Database: The database will also remain in a state of standby, as its action will be paired with the server's. The server will maintain the validity of the database.

5 Policies and Tactics

5.1 CODING GUIDELINES AND CONVENTIONS

Strict coding guidelines will be used in the implementation of the Money Hub System. The reason being it allows for more rapid and effective development. For the development of the C# and Java components of the system, several guidelines will be followed and discussed in this section.

The coding convention and their benefits are as follows:

- Naming Conventions- create a consistent look in the code. Allows developers to focus on the content and not the layout.
- Layout Conventions – Enables better copying, changing, and maintaining of code.
- Commenting Conventions – Enables testers and developers to better understand how the code works. Enables rapid bug fixing as well as enables non-developers to understand how the code is working

Security is also a primary concern for the Money Hub Application so a few security guidelines will also be followed:

- Secure Resource Access – Use of a database to hold passwords and user account information rather than storing in a file that the client has direct access to.
- Security-Neutral Code – Means the program runs with whatever permission is allowed for that user when they register their account. No one can change their *own* permission level once they create their account.

5.2 PLANS FOR TESTING PRODUCT

Various testing strategies will be used at different stages of implementation in order to ensure the components are working as designed. Unit testing will be the primary method to test various cases. Since the implementation is not yet complete at this point the testing will be saved for its own report.

5.3 MAINTAINING PRODUCT

Once out of the prototyping phase with a proven application, product maintenance will be essential to keeping users' level of contentedness high. Additionally, any bugs in the system will need to be periodically fixed once they are found. The system will have means by which the users can communicate any errors in the system that they encounter.

5.4 ORGANIZATION OF CODE

Organization of the source code shall be in a project files directory. Within that directory, there will be additional directories for the user interface as well as one for both the server and database file. This organization strategy keeps the components and dependencies for the system all in one place which allows for quicker development.

5.5 GENERATING SYSTEM DELIVERABLES

In order to ensure the application and each of its components are compiled correctly, a batch script seems like a valid option to set up the various dependencies in the system. A detailed user manual will be developed prior to the final delivery to ensure the evaluator can correctly use the system.

6 Detailed System Design

6.1 CLASSIFICATION

6.1.1 – Client Component

Files associated with a visual studio project will be included, i.e. resource files, .config files, and properties for the project. Additionally, the files that are part of the client will contain all the code that runs and displays the GUI.

The following are the classes and types of the project so far:

- accountCreator *form*
- Navigator *form*
- AccountTypeEnum enumeration
- AdminAccount class
- Login *form*
- MoneyHubClient *form*
- UserAccount class

6.1.2 – Server Component

- ServerDriver class
- MySQL JDBC library

6.1.3 – Database Component

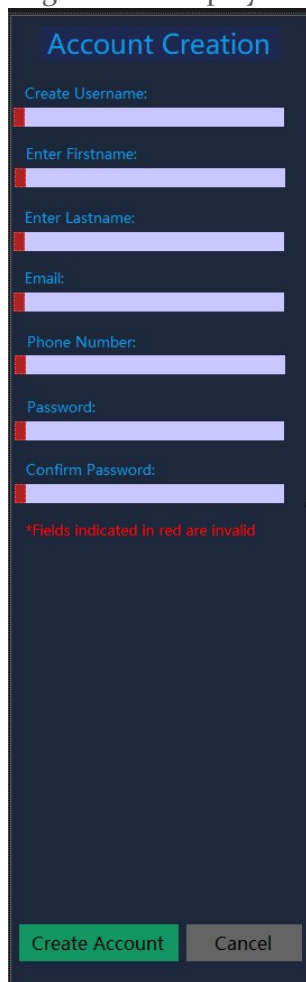
- MySQL server

6.2 DEFINITION

6.2.1 – Client Component

The client contains many classes and forms that handle the users actions while interacting the system.

-accountCreator: This is the form that is displayed while a new class is being created. If a field is invalid, it will be indicated with a red marker and the overall error message will be displayed.



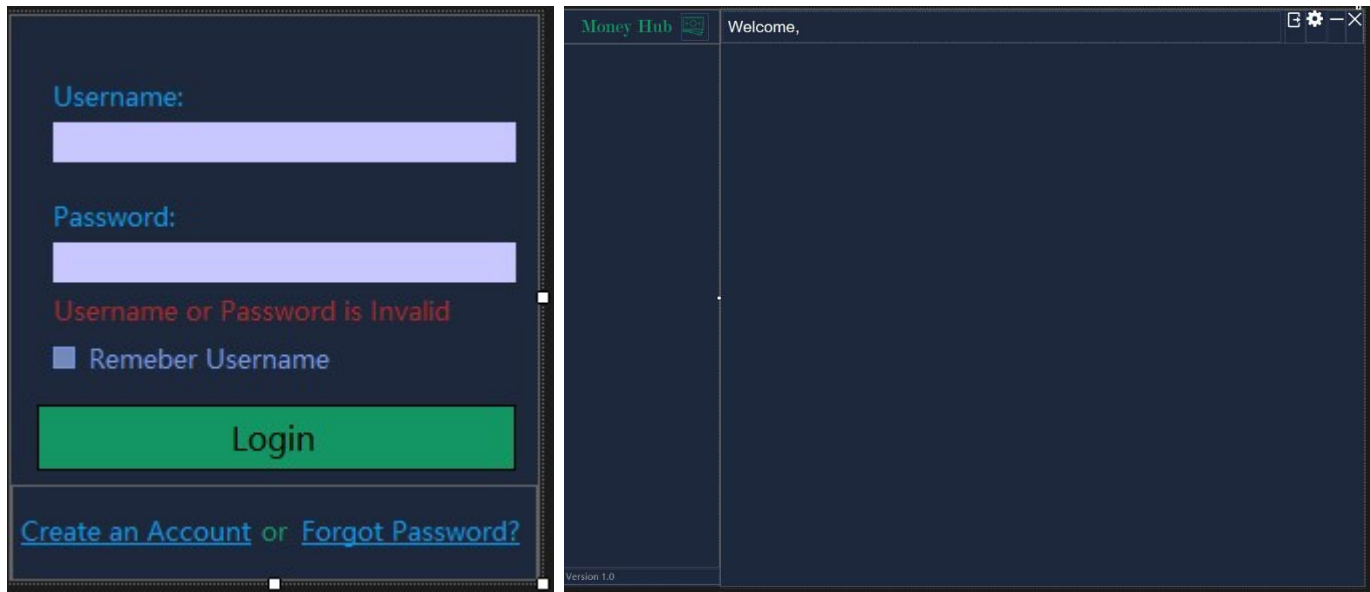
The 'Account Creation' form is a vertical layout with a dark blue background. It features several input fields, each with a red vertical bar on the left side, indicating they are required or invalid. The fields are labeled: 'Create Username:', 'Enter Firstname:', 'Enter Lastname:', 'Email:', 'Phone Number:', 'Password:', and 'Confirm Password:'. At the bottom, there are two buttons: 'Create Account' (green) and 'Cancel' (grey). A red error message at the bottom reads: '*Fields indicated in red are invalid'.



-Navigator (above right) – this is the preliminary design of the menu that the users will utilize to navigate throughout the application

- AccountTypeEnum- this enumeration will help classify accounts so that the forms layout can be adjusted accordingly

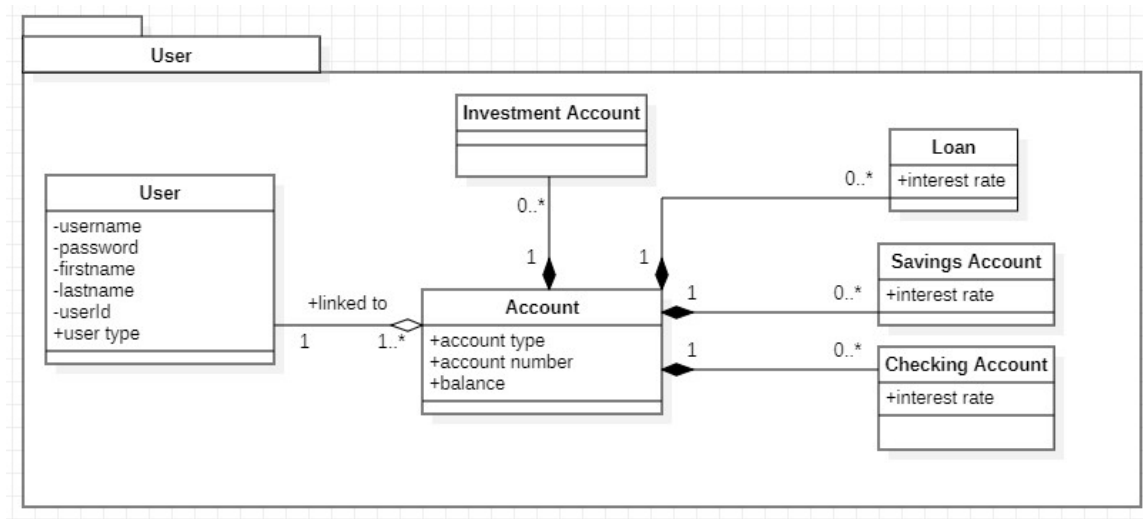
-Login (bottom left) – The first panel the users will see when they open our application. This is where the user will enter their username or password before continuing into the application. The user will also have the option to select “remember username”, “create an account” or “forgot password”



-MoneyHubClient (above right) – this is the base form of the GUI. It will hold the panels for the login as well as the navigation panel and account creator when those options are selected. Options for the settings, minimization, and exiting are in the top right of the panel. Additionally, the log out button will be visible once users successfully are logged into the system.

-UserAccount – this class will be used to store the information regarding a user’s accounts as well as other information.

The UML class diagram below will show how the user and their accounts are related:



6.2.2 – Server Component

- ServerDriver: The main application of the server. This is what will wait and listen for incoming requests from the client.
- MySQL JDBC library: A collection of classes that allow java applications to interface with MySQL database systems.

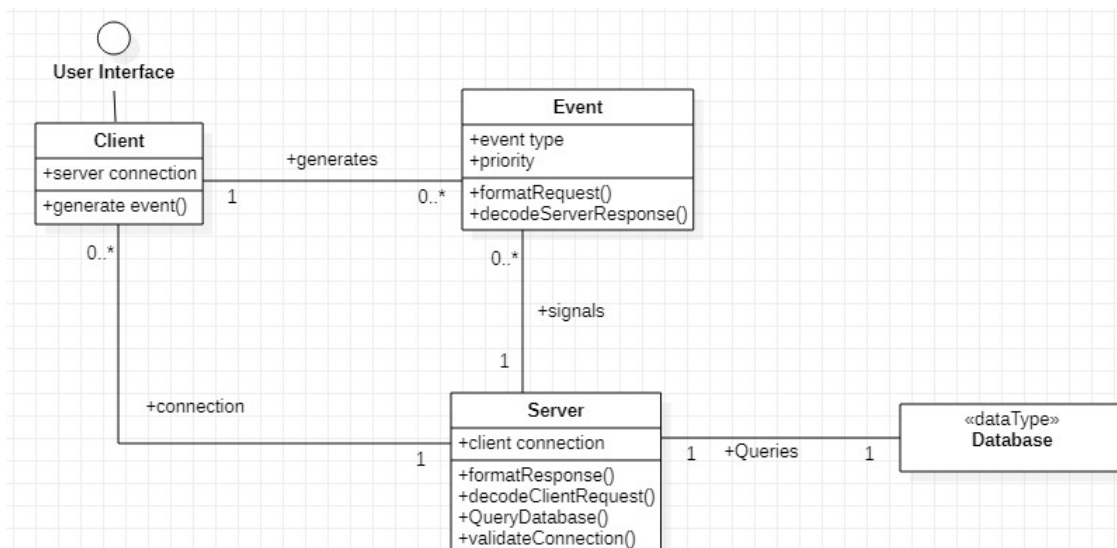
6.2.3 – Database Component

- MySQL server: The database system that is used to implement and maintain user account information

6.3 RESPONSIBILITIES

6.3.1 – Client Component

The client has the overall responsibility of connection interpreting the user's actions into events that the server and database will understand, and then be able to display back the results for that user.



6.3.2 – Server Component

- ServerDriver: Listen for request from clients, process their requests, and return either a fail state report, or the data requested by the user.
- MySQL JDBC library: Allow interfacing functionality between the server and the MySQL server.

6.3.3 – Database Component

- MySQL server: Store user information.

6.4 CONSTRAINTS

6.3.1 – Client Component

-Login – usernames and passwords must be retrieved and validated from the database before they can be logged into the system. This will require a connection to view the user's information. Furthermore, this will require more time to be spent validating the credentials

6.3.2 – Server Component

- ServerDriver: Must allow for concurrent access to the database.

6.5 COMPOSITIONS

6.3.1 – Client Component

The client is composed of various forms that allow for the display of various windows while using the product. These components were discussed in section 6.2 and will be added to as needed as development progresses. An example of a component that has not yet been implemented will be the overview window. This form will be where the summary of the user's data will be presented.

6.6 USES AND INTERACTIONS

6.3.1 – Client Component

-This component will interact with the server and database components via data sent over a TCP connection.

6.3.2 – Server Component

- Interacts only with the client and the database server

6.3.3 – Database Component

- Interacts only with the server application

6.7 RESOURCES

6.3.1 – Client Component

-GUI will require the most memory since the forms will utilize various images held in the resource folder of the program.

6.3.2 – Server Component

- ServerDriver: Minimal memory usage. Each concurrent access must have access to the database server. This is a possible race condition and must be mitigated by locking the database connection to a specific thread.

6.8 PROCESSING

6.3.1 – Client Component

-Logins are validated by sending the username and password entered in the fields of the login form to the server. The server will then query the database. If the username exists and the entered password matches that which is held in the database, the login will be accepted.

-Creating an account will require all the fields in the account creator form to not be empty as well as meet the requirements laid out in the requirements document.

-Once a user's account is created, the data will be entered into an object that is then formatted and sent to the server where it is parsed and stored in the database. The same process but in reverse is used when the user is logged in and the data in the overview form is loaded.

6.3.2 – Server Component

- ServerDriver: Will spawn a new thread for each concurrent access to the database. This will be limited the number of threads that can be executed by the system's processor.

6.9 DETAILED SUBSYSTEM DESIGN AND INTERFACES

6.3.1 – Client Component

The client has the main service of retrieving, setting, and displaying data from the database. Each user will maintain a data structure that contains all their account information. Once the user logs in successfully, since efficiency is not an enormous concern for the prototype phase, the data for each of the user's accounts is loaded into a data structure. From there, the client will parse the data into the various forms that the user can view.

The major exceptions that can affect the usage of the system are outlined in the table below.

Exception Type	Description
Login exception	User credentials are invalid
Server Connection exception	The connection between the client and the server is not valid
Data Format Exception	The data being returned from the client or from the server does not have the correct format and is being rejected.

The client will contain subroutines that create events, format data, and unpack data to be displayed in the GUI. The parameters for the format data and create event subroutines will depend on the type of subroutine being performed. The unpack data subroutine will be called in response to an event that requires data returned from the server.

6.3.2 – Server Component

- Terminal Interface only. Minimal human interaction possible. All interaction will be carried out through software.