CSCI441_VA Software Engineering

Group 4

InvestME

Github Repository

Kristyna Mason, Nicholas Heyd, Jeremiah Warber, and John Bordelon

Table of Contents

Customer Statement of Requirements

1.1 Problem Statement

Why do people spend their hard-earned money on frivolous things? After all, it is not like everyone is rich. Most of us have to worry about our finances and try with the best of intentions to increase our savings. For some reason, this simple concept is deceivingly difficult to achieve. It takes a great deal of self-discipline to responsibly manage personal finances. We, like many others, spend money on frivolous items, which in hindsight shows us that we don't want to continue purchasing unnecessary goods and services. If we had a tool to help us save money or invest it properly, it would undoubtedly help with this problem. We hope for a software application that encourages people to stop spending money on unnecessary things and instead, save their money. We want to see our savings grow, and by doing so, we feel this will increase our financial wealth and freedom. As customers, we are asking developers to create such a software application.

Recent studies have shown how Americans, specifically, spend money on nonessential items. It is alarming to know this dollar amount gets up to tens of thousands a year (Backman, 2019). Typically, the average American does not have a lot of money in their savings accounts (Backman, 2018). Why does this problem exist? According to a report on The United States of Financial Waste, conducted by Hloom, going out to eat is the biggest contribution to spending money on nonessentials (Martin, 2019). Perhaps people simply grip to the joy of going out to eat with friends or family. Spending money on things that bring us joy is not necessarily a bad thing; however, we may often find ourselves doing this more than we should.

The report by Hloom shows millennials are the biggest contributors to overspending on nonessentials (an astonishing 72.36%). Not only this, but according to the report findings, there is a trend where the newer generation spends more on going out to eat than the previous generation, comparatively. Alcoholic drinks and coffee, for instance, often are considered nonessential expenses and quickly add up. Many of us love coffee and seemingly need to have a cup of it every morning. However, this daily consumption can quickly become a costly routine purchase. A Starbucks latte can easily be just under $6. Now, imagine spending $6 a day, 365 days a year; this equates to $2190 annually on coffee alone! Buying expensive coffee is a great example of frivolous spending on unnecessary items. Spending money on nonessentials is a problem we wish to solve. We need to be reminded of how to better spend and budget money.

There should be a feasible way to encourage us to make better purchases. Conveniently being able to keep track of expenditures would be a logical start. Now, there are plenty of smartphone applications on the market that allow us to see where our money is going. However, we want the developers to produce something that offers more. Simply having this visibility is not enough. We need alternative options where we could spend our money, options such as stocks; displaying options would help us practice doing the right thing with finances, and at the very least, provide more awareness. For example, we often hear about people making money all the time by investing in the stock market. Who would not love to hitch a ride on this bandwagon; but where do we start? Some may know a little about the stock market, but often people would not know what to buy. Unfortunately, there is very little formal education we receive regarding investing. Moreover, people may find it difficult to break the habit of spending

money on nonessentials. For example, we know if our friends send a text message asking to go have a craft beer, or a video game we have been waiting to purchase is being released, it can be hard saying no. However, having an application show us an alternative that could possibly help us create savings by means of investing. It will help our users stay motivated to continuously make better financial decisions.

For security, time, and liability reasons, we do not want the application linked to any personal bank accounts. Therefore, a software application to help save money would require us to input financial transactions; something like this needs to be very quick, easy, and intuitive. If it is difficult, or too time consuming, we will most likely not do it for long. Inputting expenditures will be one of the most common functions we will use, so it is critical this function works. There should be a feature to help remind us to input expenditures. For example, when we walk into one of our favorite restaurants, the application should acknowledge our location. Then, the application will send a reminder to us to ensure the purchases were inputted. Of course, we would like to be able to turn these notifications off at any time. After we input an expenditure, the application should show the various stocks we could have purchased instead.

Being able to view the overall history of our spending would be essential. What would be even better is if we could view our spending history during a specific date range. This would allow us to break down our spending habits and provide a better understanding about our finances. Another function we would like the software application to have is to provide a summary of potential gains. For example, imagine we enter a specified date range for the expenditures in that timeframe. The application will display the total dollars spent while breaking down each transaction. At the same time, the application should display how much we could have earned if we invested in actual stocks! As users, this would be extremely motivating. Ostensibly, the point of this application is to educate us on the stock market. It would be great then if we were able to search for a stock using its ticker symbol.

Providing us with a summary of potential earnings would be one of the greatest features of the software application. Let's say the application showed we spent $500 on going out to eat last month. If we saw that $500 could be increased to $900 just from investing, it would be shocking. More money is a great motivator for self-improvement, and seeing this data would help us stay focused on our goals. Additionally, having a timestamp for each logged expenditure would also be a necessary feature for the application. This will allow for the total amount spent in a specific range of time. We would also like the developers to provide a feature where the application furnishes a link to a brokerage such as Robinhood, or TD Ameritrade. This way, if we wanted to actually purchase the stocks being presented, we could. To help with this feature, we would like it if the user can search for more information about the stocks being presented to them. For the application to provide accurate stock proposals, it will need to automatically update historical, and existing stock data.

After we input data, we should be able to clear history, edit, or delete previous expenditures. This process needs to be easy and intuitive. Upon creating a new account, we should be able to use our email as a username. This will allow a sense of convenience. Additionally, we could have the option to sign in with a valid social media account, or public email account. We want the account creation process to feel intuitive. There are also many applications and websites who offer these services, and so should this application.

There are some standards in information security we would like the application to adopt. For example, upon creating a new account, we would like to have common security features such as only allowing a password to contain no less than eight characters, have at least one uppercase letter, and include a special character. This is a standard practice for creating a secure password. Additionally, we want to be able to change our password anytime. This criteria for password management will help our accounts stay secure. Deleting our accounts should also be a convenient process, and it will help in securing our information. We would like the software developers to implement user accounts referencing a unique user identification number. If a we should choose to delete an account, the application will completely delete all data associated with that specific account ID.

We would like all our data from the application saved to the cloud. This includes the saved data from expenditure inputs, as well as user profile information. Cloud computing is becoming increasingly popular, and we have come to expect it in software applications. There is a fun feature we would like the developers to implement, and that is having a badge or trophy function. Obtaining badges in the application will allow us to compete with friends and give a sense of fun in making better financial decisions. For example, a simple badge could be earned when we create a new account or for our first expenditure input. We want the developers to have fun with this feature, and be creative with the various achievements or goals.

Having daily notifications is common in current applications. This application should be no different - meaning it will provide daily notifications to us. Again, this feature will be optional. To help make sure we use the application properly, we ask the developers to please implement a tutorial. As a new user, imagine downloading this application and have no idea clue about investing; please make sure we have a guide to ease us into the process. The tutorial will recognize us as a new user and give us the assistance we need.

Overall, data shows spending money on nonessentials is an increasing trend, and is a problem, specifically, for Americans. Having a tool to remind us how we could better spend our money is a way to help remedy this issue. We are asking the developers to implement this tool as a software application for smartphones. The application, along with all of its intuitive features, will help us make better financial decisions, and increase our financial freedom.

1.2 Glossary of Terms

API: A set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, or other service.

Backend: Part of a computer system or application not accessed directly by the user, typically responsible for data storage and manipulation.

Better purchase: An expenditure that is not defined as unnecessary or frivolous. Specifically, a type of investment.

Brokerage: A business that acts as a “middleman” between a buyer and seller of stocks.

Cross Platform: Application that is able to be used on different types of devices.

Expenditure (financial transactions): Any money spent.

Finances: The management of a person's money.

Financial Decision: A choice of where a person chooses to invest his/her income.

Financial Wealth: The total sum of all of a person’s assets, savings, and investments.

Flutter: Google's open-source* mobile application development framework. It is used to create Android and iOS apps and is the main technique of generating Google Fuchsia apps.

Frivolous Item (nonessential item): Purchases that are unnecessary or irrelevant to a person’s goals in life / self-indulgent expenditures.

Frontend: The computer system component or application that the user directly interacts with.

Invest: Put money aside with the intent of achieving a profit through financial designs and arrangements.

Life Cycle: A structure followed by a development team. It consists of a detailed plan that describes how to develop, maintain and replace specific software components.

Open-source:

Overspend: A person spending more than his/her budget allows, or spending an amount equal to or over his/her total income without regard to savings.

Retention: After establishing a ‘churn time’, find out how often users are using the application.

Savings: Money put aside for future expenditures or retirement.

Savings Account: An account to put money designated for future expenses in.

Share: The equity of a company divided into units, so that a percentage of a firm can be owned by multiple people.

Stock: An investment that gives the holder a fraction of ownership of a specified company.

Security: The ability of someone to protect his/her financial privacy and income.

Spending habit: An individual’s pattern of expenditures.

Stock (Ticker) Symbol: A symbol that is a unique series of letters assigned to a company for trading purposes.

Timestamp: A digital record of a specific event's moment of occurrence.

System Requirements

2.1 User Stories

In a user story, the setting is a world in which the user interacts with the software. The story is written from the user's point of view and talks about things from the perspective of the user. The user perspective is very important because our principles and values in agile say that we are going to define our progress based on giving the user the ability to do something valuable with the software that they were not able to do before.

The user stories written below show specific cases and specifications for program functionality, as well as a weight to assess the relative time expected to take of each requirement. These features are unordered and are merely a list of the criteria of end user story and comparative weighted significance. It is essential to note that in future iterations of this report, these instances will be further discussed. From the user's point of view, the following is told as a <who> I want a <what> so that I can <why>, with the intention of fully encapsulating what the user should expect when opening and using the referenced software.

Identifier	User Story	Weight
ST-1	As a user, I want to use my email or sign in with a social media account to sign-up, so I can create an account.	9 points
ST-2	As a registered user, I want to change my password so I can keep my account secure.	4 points
ST-3	As a registered user, I want to reset my password so I can keep my account secure or if I forget my log-in information.	4 point
ST-4	As a registered user, I want to delete my account so I can keep my information secure.	4 points
ST-5	As a registered user, I want to view the terms and conditions so I can see how my data is being used.	2 points
ST-6	InvestME should give badges upon certain criteria met, e.g. creating an account, inputting first expenditure, looking at stock market data, comparing stock market data.	6 points
ST-7	As a registered user, I want to view a tutorial so I can understand how to input an expenditure.	4 points
ST-8	As a registered user, I want to add, edit, and delete the amount, the location, and a personal note on InvestME, so that my information remains up-to-date.	4 points

ST-9	As a registered user, I want to save all my data to the cloud so I can access it from another device.	6 points
ST-10	As a registered user, I want to view a tutorial so I can understand how and why I should track my previous expenditures.	2 points
ST-11	As a registered user, I want to track my previous frivolous expenditures on InvestME, so I can see which investments would benefit me.	3 points
ST-12	The application should automatically update specific historical and real-time stock market data.	5 points
ST-13	As a registered user, I want to view a tutorial so I can understand how to view and analyze historical and real-time data and apply it to my entries..	3 points
ST-14	As a registered user, I want to view historical and real-time stock market data of any registered company on the application, so I can apply that information to my data.	7 points
ST-15	As a registered user, I want to view a tutorial so I can understand how to search stocks by their ticker symbol or registered company name.	2 points
ST-16	As a registered user, I want to search for various stocks by their stock (ticker) symbol or registered company name, so I can find the companies easily.	3 points
ST-17	As a registered user, I want to view a tutorial so I can understand how to compare a given date range to specific stock prices.	4 points
ST-18	As a registered user, I want to compare a given date range of my previous frivolous expenditures, with specific stock prices given that range, so that I can see what I would have made.	9 points
ST-19	As a registered user, I want to view a tutorial so I can see how I am able to purchase stocks outside of InvestME.	2 points
ST-20	As a registered user, I want to request more information on how to purchase the stocks I have compared, so that I may consider purchasing them outside the application.	4 points

ST-21	As a registered user, I want to view the amount of money I could have earned over a specific time frame if I compare it to a stock had I invested instead of spending the money on something else.	8 points
ST-22	As a registered user, I want to choose to receive a notification to remind me to log my expenditures when I enter a common location where I may spend money frivolously.	5 points
ST-23	As a registered user, I want to choose to receive daily notifications, so I may be reminded to input frivolous expenditures.	2 points

2.2 Non-functional Requirements

A non-functional requirement is described as a requirement that sets out criteria that can be used to judge a system's operation rather than its specific behaviors.

2.2.1 Functionality

To assist the user with signing up for an account in the InvestME the user will be allowed to sign in using their existing social media accounts or public email address, without divulging their credentials to the InvestME app.

If the user would rather create a user account instead of signing in with their social media or public email address, an account can be created with a signup process. The user would need to enter their name, email address, and a password that meets a specific set of guidelines to secure the account. I

2.2.2 Usability

The application should focus on providing the user with a clean, intuitive, and visually appealing interface. This will be made possible by using Flutter as the development platform. Flutter allows for the creation of widget interfaces that allow for the same experience regardless of the device being used.

2.2.3 Reliability

To maintain reliability the user will receive a confirmation message whenever a stock is added to or removed from their list. If an error occurs, the user will receive a notification with an error message and timestamp of the event. On the backend, the database will be replicated to ensure that data is recoverable in the event of corruption or errors.

2.2.4 Performance

In order to maintain optimal performance across multiple platforms, the application will need to be as lightweight as possible. During the development of the application Flutter should be used as a development platform, this allows for cross platform compatibility. Rather than using platform specific functions and buttons, Flutter uses widgets on top of the hardware to make designing the application more efficient and lightweight.

For the application to be efficient, all tasks initiated by the user should be completed in a timely manner. The user should be able to sign in, as well as retrieve their account data from the database in a timely manner. The database should be able to serve several users without delay or corruption.

2.2.5 Supportability

The frontend of the InvestME app will support mobile devices such as Android and iOS, as well as have a web interface for use in a browser. A user will be able to manage their account from the available interfaces. From the interface the user will have the ability to change and securely delete their account from the database.

2.2.6 Data Integrity

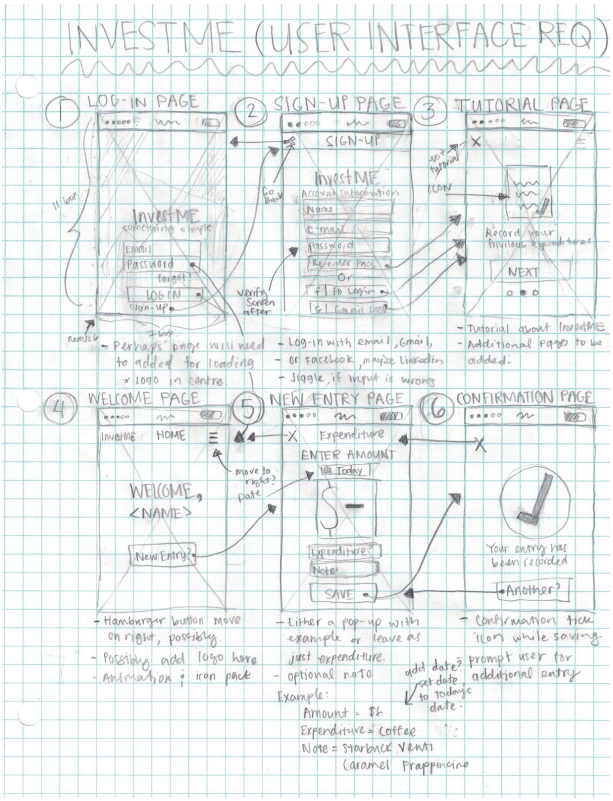
To ensure accurate and up to date data, an API that provides real-time and historical data for the various companies on the stock exchange should be used. By having accurate data, this will allow the user to make an educated decision on which companies to follow on the stock exchange. The investment app will not modify any of the data received or processed by the API queries.

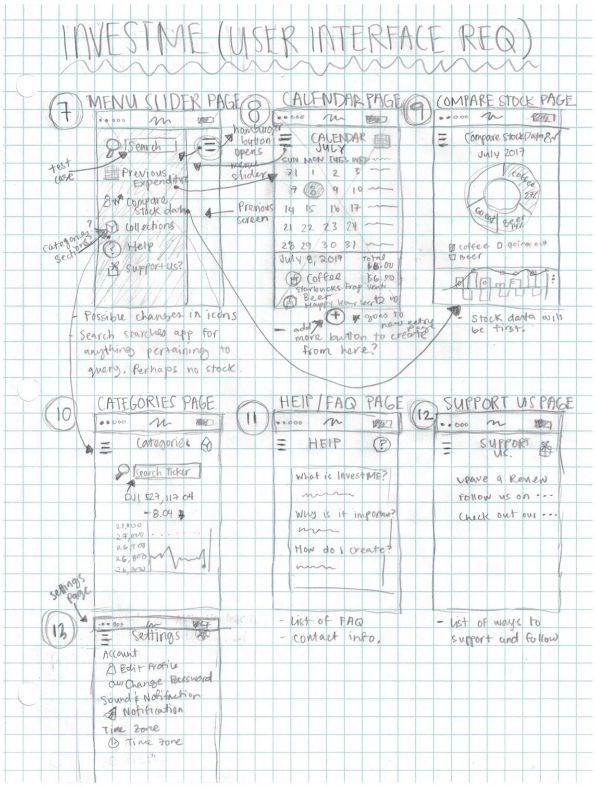
2.2.7 Maintainability

Software maintainability is defined as the degree to which an application is understood, repaired, or enhanced. Over the course of the development life cycle the development team will perform maintenance on new deployments or enhancements of the InvestME app. The maintenance performed is required to keep the source code readable and to allow for proper documentation.

2.3 UI Appearance Requirements

Below are the basic rough UX designs to help us understand the functionality and navigation methods needed for InvestME.





Functional Requirements Specification

3.1 Stakeholders

The term "stakeholder" can loosely be defined as anyone with an interest in the project.

- **Users**
    - The user is at the forefront of the development team's mind. The application is specifically designed with an intent to address a very basic concern: learn how to spend money better. In this context, the phrase "learn how to spend money better," means juxtaposing the value of investing and frivolous spending. The user should be operating the app daily, since his/her consistent use will provide a lesson in investing. By working with the app, the user will become familiar with the swings of the stock market, the alarming rate at which senseless spending adds up, and a novel perspective toward the power of investing.
- **Professor**
    - Dr. Anas Hourani is the arbiter over the course, and, therefore, this project for CSCI441. The project is designed and developed with minimal supervision, but with the approval of Dr. Hourani. He has created the structure, format, and due dates for the project. The project must meet the standard put in place by him and be formally graded upon completion.
- **Developing Team**
    - All four team members began developing this application as part of an assignment in CSCI441, but also with the understanding that the app would fill a need in their lives. During brainstorming, a conclusion was drawn that all team members valued the importance of education and investing. The team decided that a way to combine these two passions would not only be a personal boon, but beneficial to anyone else wanting to learn.
- **Project Testers**
    - The team members have several people, who are not part of the development team, that will be testing the project and providing feedback. In some cases, the tests will consist of individuals with a technical background to gain another opinion of the application; in other instances, a tester may be someone completely outside the technical field to best garner an outsider's perspective.

3.2 Actors and Goals

Actor	Actor's Goal	Type	Use Case Name
User, Database	To create an account using my email, or existing social media Account.	Initiating	CreateAccount (UC-1)
User	To change my password.	Initiating	ChangePassword (UC-2)
User	To reset my password.	Initiating	ResetPassword (UC-3)
User	To delete my existing account.	Initiating	DeleteAccount (UC-4)
User	To view terms and conditions.	Initiating	ShowTermsConditions (UC-5)
System	To give badges for user achievements and meeting criteria.	Participating	GiveBadge (UC-6)
User	To view the input expenditure tutorial.	Initiating	InputTutorial (UC-7)
User	To add an expenditure, location, and personal note.	Initiating	AddExpenditure (UC-8)
User	To edit an existing expenditure, location, or personal note.	Initiating	EditExpenditure (UC-9)
User	To delete an existing expenditure, location, or personal note.	Initiating	DeleteExpenditure (UC-10)
User	To save all user data to the cloud.	Initiating	SaveData (UC-11)
User	To view the track expenditure tutorial.	Initiating	TrackExpenditureTutorial (UC-12)
User	To view previous expenditures.	Initiating	ViewExpenditure (UC-13)
System, Database	To automatically update historical and real-time stock market data.	Participating	UpdateMarketData (UC-14)
User	To view the market data tutorial.	Initiating	ShowMarketDataTutorial (UC-15)

User, Database	To view historical and/or real-time stock market data.	Initiating	ViewMarketData (UC-16)
User	To view the search stock tutorial.	Initiating	SearchStockTutorial (UC-17)
User, Database	To search for a specific stock by the stock (ticker) symbol.	Initiating	SearchTickerSymbol (UC-18)
User	To view the compare-expenditure tutorial.	Initiating	ShowCompareExpenditureTutorial (UC-19)
User	To compare expenditures in a specified timeframe with stocks, also in a specified timeframe.	Initiating	CompareExpenditure (UC-20)
User	To view the purchase-stocks tutorial.	Initiating	ShowPurchaseStocksTutorial (UC-21)
User	To view additional information on purchasing stocks.	Initiating	InquireStocks (UC-22)
User, Database	To view potential earnings in a specified timeframe.	Initiating	ViewPotentialEarnings (UC-23)
User	To turn on expenditure reminder notifications.	Initiating	TurnOnDisplayExpenditureReminder (UC-24)
User	To turn off expenditure reminder notifications.	Initiating	TurnOFFDisplayExpenditureReminder (UC-25)
System	To provide user expenditure reminder notifications.	Participating	DisplayExpenditureReminder (UC-26)

3.3 Use Cases

3.3.1 Use Case Casual Description

UC-1: createAccount

Allows User to register a social media account or a public email address on an application serve to create an InvestME account. This will allow new users of the InvestME app to easily log in to the app using their existing Facebook or Gmail account.

Derived From: ST-1

Note: A User will not need to reauthorize their account if logging into the application from the mobile device/ computer they had previously logged in from.

UC-2: accountModification

Allows registered User of the InvestME application to change their account password in order to keep their account secure.

Derived From: ST-2

UC-3: accountPasswordReset

Allows the User to reset the password of their account if they are unable to log in due to a forgotten password, the user will be able to reset the password of the account.

Derived From: ST-3

UC-4: accountDeletion

Allows a registered User to securely delete their InvestME account in the event that they no longer wish to use the application (<<extend>> UC-2: accountModification).

Derived From: ST-4

Note: The User should receive a notification confirming that they wish to delete their InvestME account. Upon successfully deleting their account the user should receive an email confirming the deletion of the account.

UC-5: termsAndConditions

Allows a registered User to view the terms and conditions of using the InvestME app and how their data will be used (<<extend>> UC-1: Account Creation).

Derived From: ST-5

Note: The terms and conditions shall be presented at the time of account creation as well as on request via a “Terms and Conditions” link.

UC-6: achievementsAndBadges

Allows the InvestME app to give badges based upon the user meeting certain milestones. Badges can be earned by signing up for an account (<<extend>> UC-1: Account Creation), inputting expenditures, etc.

Derived From: ST-6

UC-7: viewTutorials

Allows User to view tutorials on how to perform certain tasks within the InvestME application. The user will have access to how to use the application, how and why to monitor past expenses, how to view and evaluate stock information (historical and real-time), how to search for stocks based on ticker symbols, how to compare stocks given a particular range, and how to buy stocks outside the InvestME application.

Derived From: ST-7, ST-10, ST-13, ST-15, ST-17, ST-19

Note: Tutorials shall be presented after the account has been created. There should be an option to skip the tutorial. The tutorials shall have their own section to refer to specific topics at a later date.

Note II: Stocks can not be purchased in the context of the InvestME application. A link to an outside broker may be established.

UC-8: dataInput

Allows the User to update their personal expenditures in the context of the InvestME application. The user will have the ability to add, edit and delete the amount of each expense.

Derived From: ST-8

Note: This feature allows the user to keep their account up to date.

UC-9: cloudStorage

Allows the User to sync their InvestME account to the cloud. This will allow users to access their account data across a number of different platforms.

Derived From: ST-9

UC-10: trackExpenditures

Allows the User to track previous expenditures in the InvestME application.(<<extend>> UC-7: DataInput).

Derived From: ST-11

UC-11: systemRecommendations

Allows the InvestME app to make recommendations based on the input of previous expenditures by the user (<<extend>> UC-9: TrackExpenditures).

Derived From: ST-11

Note: The applications should make recommendations that benefit the user.

UC-12: viewData

Allows the User to view the historical and real-time data on the stock market of any registered company.

Derived From: ST-14

Note: The data is provided to the user via the Alpha Advantage API.

UC-13: searchData

Allows the User to search for companies in the InvestME app with their ticker symbol or registered company name.

Derived From: ST-16

UC-14: compareExpenditure

Allows the User to compare the date range of expenditures previously entered and compare it to the date range of the specific stock price. The user will see how much could have been earned if the money had been invested from the expenditure instead of being spent on something frivolous.

Derived From: ST-8, ST-9, ST-18, ST-21

UC-15: saveData

Allows the User to save the information searched to their InvestME account in order to apply it to the user's expenditure records.

Derived From: ST-14

UC-16: updateData

Allows the InvestME application to update historical stock data automatically and to provide data on the stock market in real time.

Derived From: ST-12

UC-17: userRequests

Allows the User to request information on how to purchase shares from companies compared to them in the InvestME app.

Derived From: ST-20

Note: Links to online brokerage companies will need to be made available to the user. Perhaps more than one option should be presented.

UC-18: dailyNotifications

Allows User to receive daily notifications from the InvestME app for input of frivolous expenses.

Derived From: ST-23

UC-19: geofencingNotifications

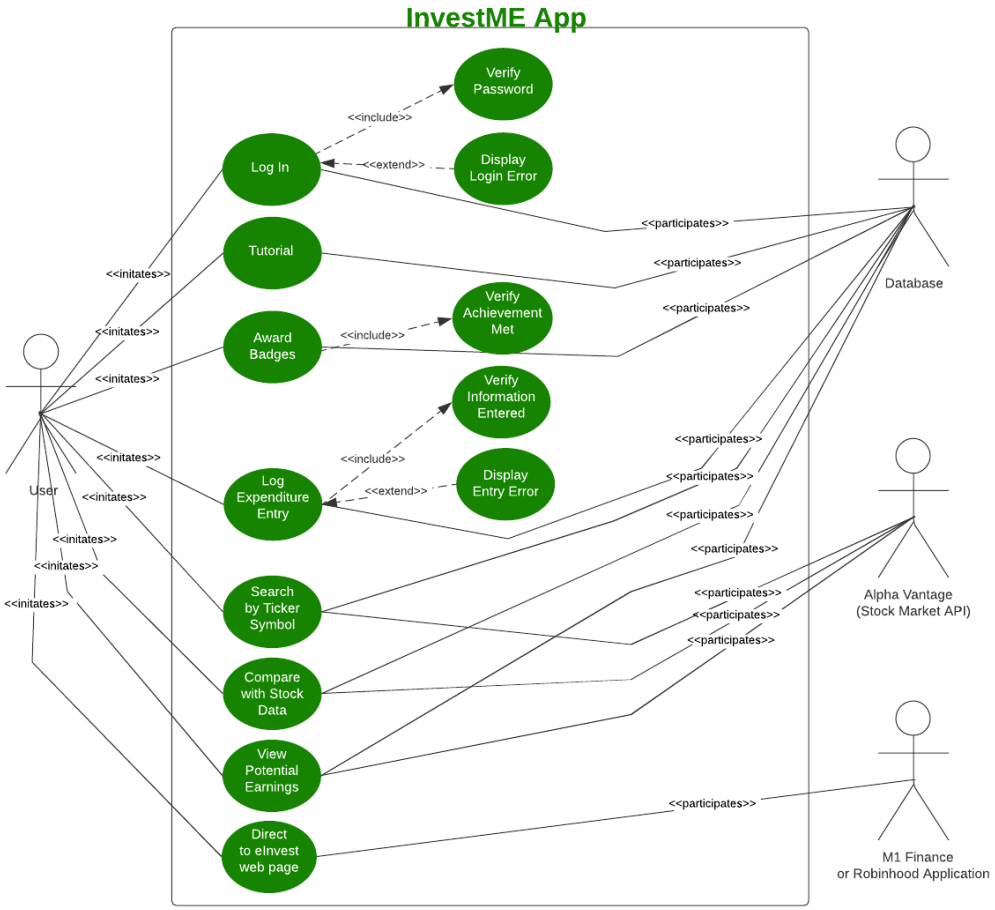
Allows the User to receive notifications to remind them of logging expenses when entering a location where they are known to spend money needlessly.

Derived From: ST-22

Note: In order for this feature to work, Geo-Fencing will need to be enabled in the app. The option to turn the feature off should be available.

3.3.2 Use Case Diagram

The diagram showcases all the use cases. The relationships are indicated by using such as <<include>> and <<extend>>, <<initiates>> and <<participates>>.



3.3.3 Traceability Matrix

Req't	PW	UC-1	UC-2	UC-3	UC-4	UC-5	UC-6	UC-7	UC-8	UC-9	UC-10	UC-11	UC-12	UC-13	UC-14	UC-15	UC-16	UC-17	UC-18	UC-19	UC-20	UC-21	UC-22	UC-23	UC-24	UC-25	UC-26
ST-1	9	X																									
ST-2	4		X																								
ST-3	4			X																							
ST-4	4				X																						
ST-5	2					X																					
ST-6	6	X					X																				
ST-7	4							X																			
ST-8	4								X	X	X																
ST-9	6	X										X									X			X			
ST-10	2												X														
ST-11	3													X													
ST-12	5														X		X										
ST-13	3															X											
ST-14	7																X				X			X			
ST-15	2																	X									
ST-16	3																		X								
ST-17	4																			X							
ST-18	9													X							X			X			
ST-19	2																					X					
ST-20	4																						X				
ST-21	8													X										X			
ST-22	5																								X	X	
ST-23	2																										X
Max PW		9	4	4	4	2	6	4	4	4	4	6	2	9	5	3	7	2	3	4	9	2	4	9	5	5	2
Total PW		21	4	4	4	2	6	4	4	4	4	6	2	20	5	3	10	2	3	4	22	2	4	30	5	5	2

For a link to the full-sized graph click here. This graph shows which ST corresponds to which UCs. In addition, the priority weight total for each UC can be found at the bottom of the graph.

3.3.4 Full-Dressed Description

Use Case UC-1:	**CreateAccount**
Related Requirements:	ST-1, ST-6, ST-9
Initiating Actor:	User
Actor's Goal:	To create an account using an email, or existing social account.
Participating Actors:	System, DataBase
Preconditions:	• User has downloaded the InvestME application. • There is no user account already signed-in. • The system displays a menu for available functions; at the initial main menu, the choices are "Sign-In" or "Create new account."
Postconditions:	• A new account is successfully created. • A badge is given to the user for completing the "Create Account" achievement.

Flow of Events for Main Success Scenario:

→ 1. **User** downloads the InvestME application, and creates a new account

→ 2. **User** (a) chooses to create a new account with existing email, or (b) uses an existing social media account

→ 3. **User** chooses a password for the new account

← 4. **CreateAccount** (a) validates the users input and creates a new account, (b) saves the account to the cloud by calling **SaveData**, and (c) gives the user an achievement by calling **GiveBadge**

→ 5. **User** successfully logs in with their account credentials, and begins using the application

Flow of Events for Extensions (Alternate Scenarios):

2a. **User** enters an invalid email, or social media account

← 1. **CreateAccount** (a) detects error, (b) signals to the actor

3a. **User** enters an invalid password

← 1. **CreateAccount** (a) detects error, (b) signals to the actor

Use Case UC-20:	**CompareExpenditure**
Related Requirements:	ST-9, ST-14, ST-18
Initiating Actor:	User
Actor's Goal:	To compare expenditures in a specified timeframe with Stocks, also in a specified timeframe.
Participating Actors:	System, DataBase
Preconditions:	• Users logged their expenditures into InvestME. • SaveData has saved all logged expenditures. • The system displays a menu for the available functions; ShowCompareExpenditureTutorial, and CompareExpenditure.
Postconditions:	• CompareExpenditure has displayed a table of user expenditures being compared to stocks in the specified timeframe. • A badge is given to the user for completing the "Comparison" achievement.

Flow of Events for Main Success Scenario:

→ 1. **User** navigates to the menu displaying **ShowCompareExpenditureTutorial**, and **CompareExpenditure**

→ 2. **User** selects **CompareExpenditure**

→ 3. **User** enters a timeframe to view the comparison

← 4. **CompareExpenditure** (a) checks the user logs in **SaveData** (b) calls data from **ViewMarketData**, and (c) gives the user an achievement by calling **GiveBadge**

← 5. **CompareExpenditure** successfully displays a table showing the comparison between logged expenditures and stocks within the specified timeframe

Flow of Events for Extensions (Alternate Scenarios):

3a. **User** enters an invalid timeframe

← 1. **CompareExpenditure** (a) detects error, (b) signals to the actor

4a. **CompareExpenditure** fails to retrieve user logs from **SaveData**, or data from **ViewMarketData**

← 1. **CompareExpenditure** (a) detects error, (b) signals to the actor, (c) refreshes, and attempts to retrieve data again

Use Case UC-23:	**ViewPotentialEarnings**
Related Requirements:	ST-9, ST-14, ST-21
Initiating Actor:	User
Actor's Goal:	To view potential earnings in a specified timeframe.
Participating Actors:	System, DataBase
Preconditions:	• User logged their expenditures into InvestME. • SaveData has saved all logged expenditures. • The system displays the available function, ViewPotentialEarnings.
Postconditions:	• ViewPotentialEarnings displays the dollar amount of potential financial earnings the user could have made had they invested their money into stocks. • A badge is given to the user for completing the "Potential Earnings" achievement.

Flow of Events for Main Success Scenario:

→ 1. **User** navigates to the function, **ViewPotentialEarnings**

→ 2. **User** selects **ViewPotentialEarnings**

→ 3. **User** enters the timeframe they wish to view

← 4. **ViewPotentialEarnings** (a) checks the user logs in **SaveData** (b) calls data from **ViewMarketData**, and (c) gives the user an achievement by calling **GiveBadge**

← 5. **ViewPotentialEarnings** successfully displays a summary of how much the user could have earned had they invested their money into stocks

Flow of Events for Extensions (Alternate Scenarios):

3a. **User** enters an invalid timeframe

← 1. **CompareExpenditure** (a) detects error, (b) signals to the actor

4a. **ViewPotentialEarnings** fails to retrieve user logs from **SaveData**, or data from **ViewMarketData**

← 1. **ViewPotentialEarnings** (a) detects error, (b) signals to the actor, (c) refreshes, and attempts to retrieve data again

3.4 System Sequence Diagram

The following sequence diagrams describe the relationships between key actors and the InvestME app. The user initiates the sequence by signing up for an account via their mobile device or via a web browser.

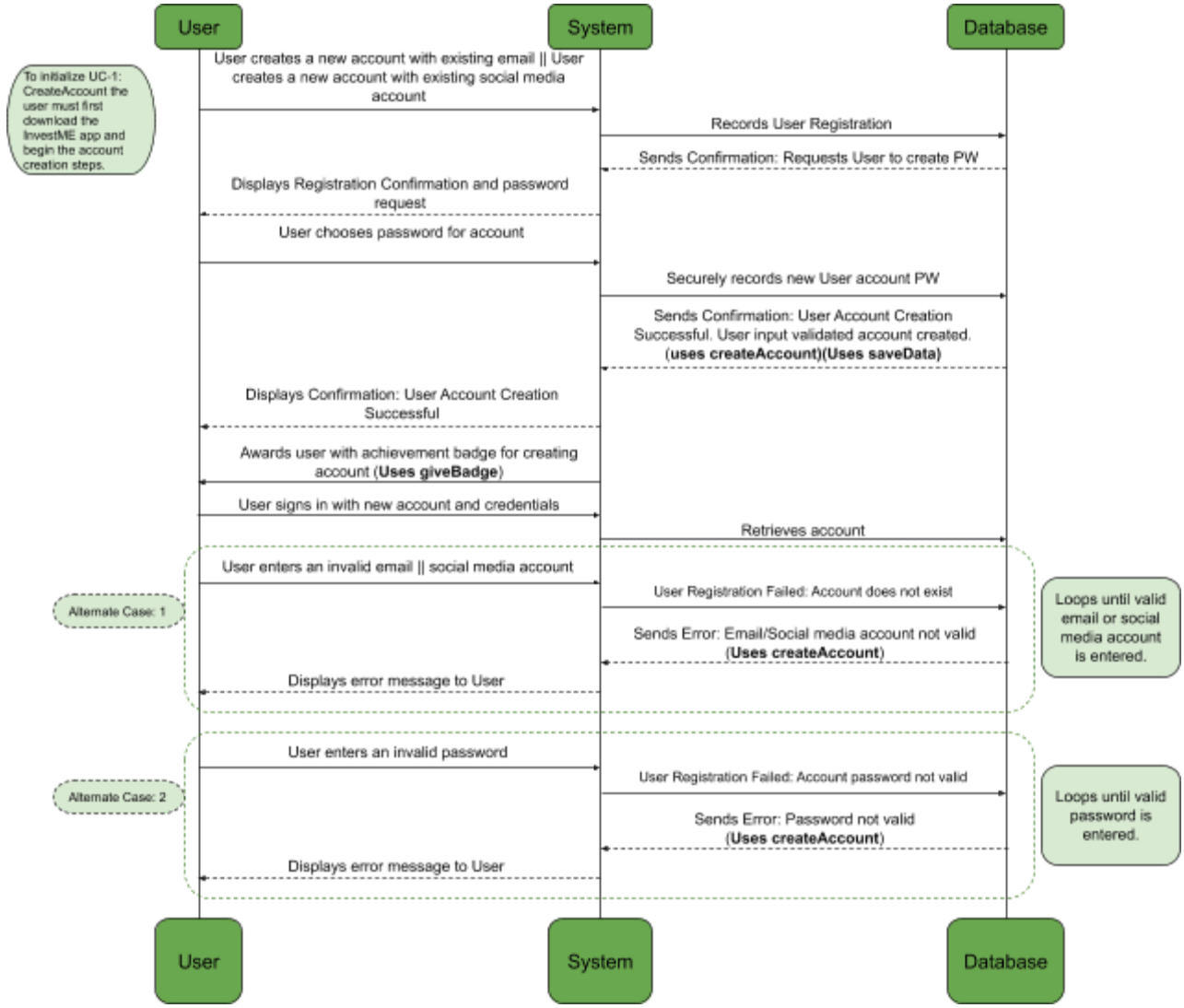


Figure 3.4.1: See UC-1: CreateAccount. When the user starts the request from a mobile device or through a web browser, they will be led to register for the InvestME account. It is necessary to create or register for an InvestME account in order to use its features. The system will take the user input data and send it to the database. The database checks for a valid email address or social media account, and sends a confirmation or failure notification to the system. The system presents the message to the user.

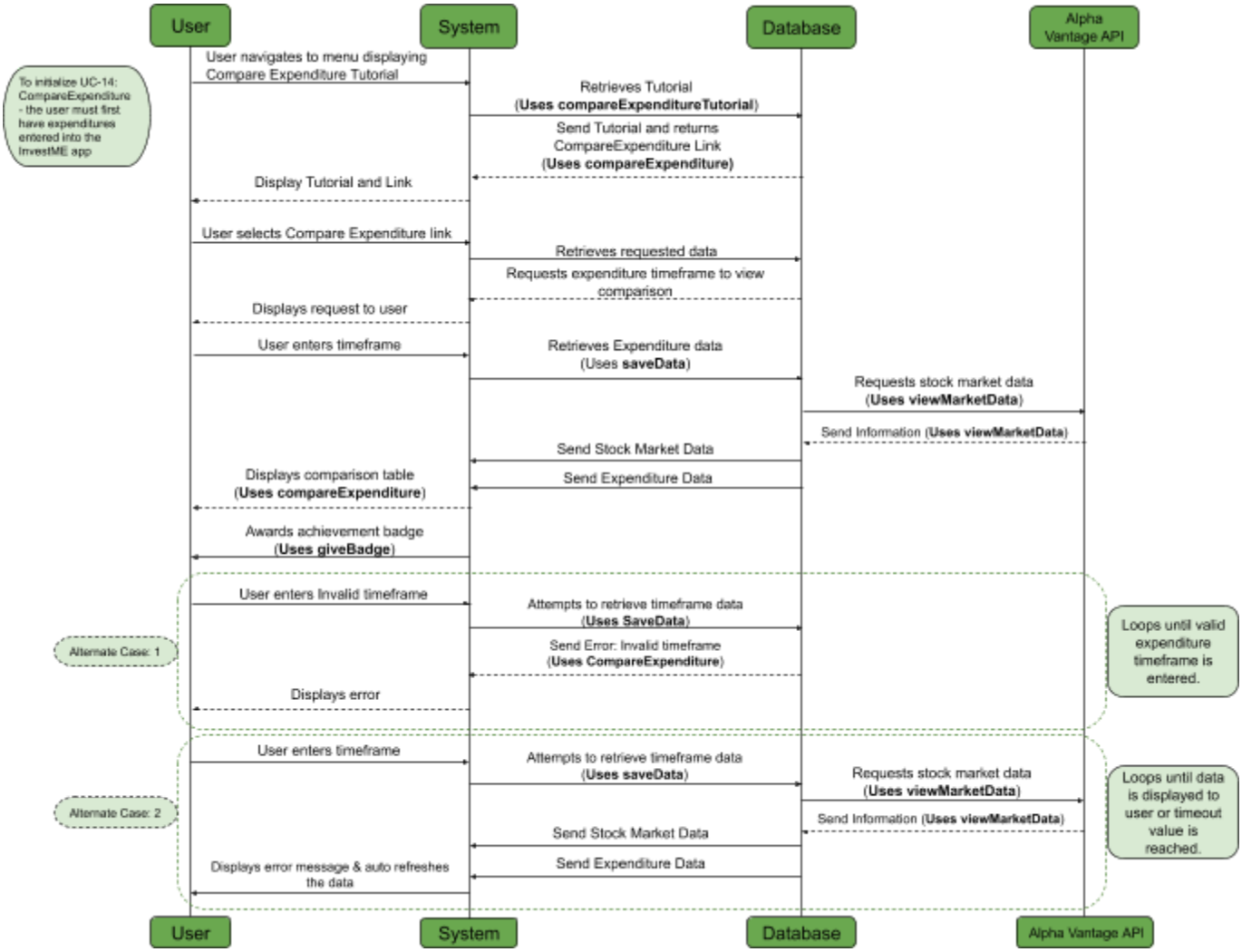


Figure 3.4.2: See UC-14: CompareExpenditure. When the user navigates to the Compare Expenditure link, this use case is initiated. The user indicates the required timeframe for the system. The system sends the query to the database and requests Alpha Vantage API stock market information. The system provides a comparative table to the user once the requested information is collected. The customer will obtain a achievement badge for market data comparison.

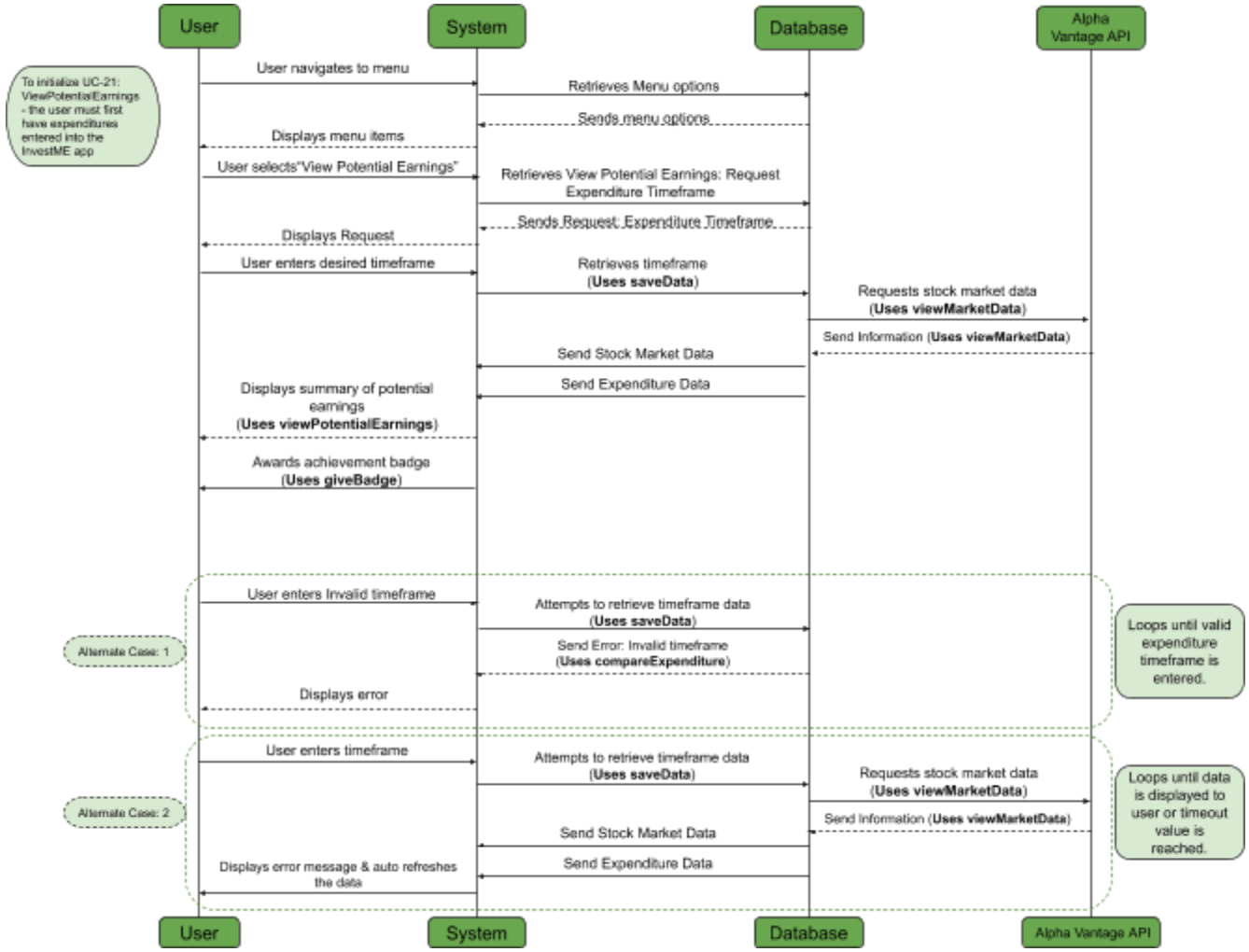


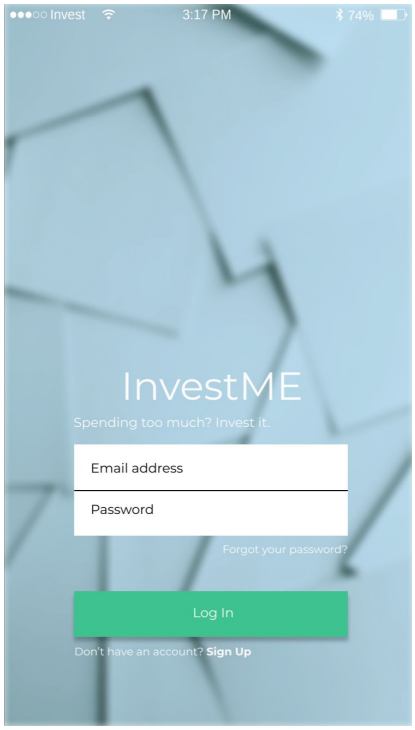
Figure 3.4.3: See UC-21: ViewPotentialEarnings. When the user navigates to the View Potential Earnings link, this use case is initiated. The user indicates the required timeframe for the system. The system sends the query to the database and requests Alpha Vantage API stock market information. The system provides a comparison between the logged expenditures and the stocks within the specified timeframe. The customer will obtain a achievement badge for completing the View Potential Earnings task.

User Interface Specification

4.1 Preliminary Design

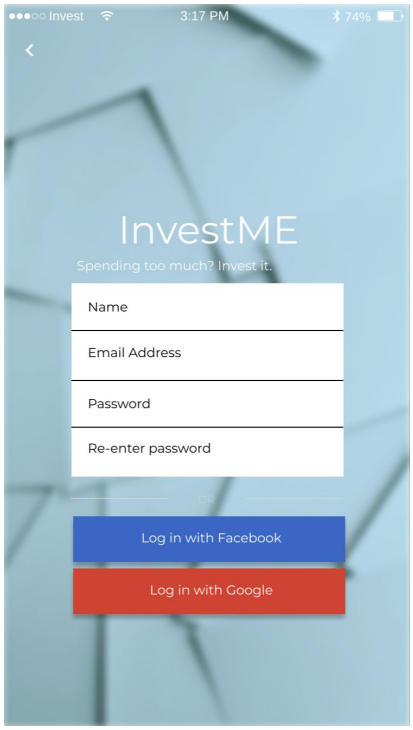
The preliminary design section showcases the many pages of the app. These pages are in chronological order to show how and explain how users enter and what each button, clickable area, and input area does. The navigational paths are as follows below and are can be seen specifically in the menu slider page.

1. Log-in Page



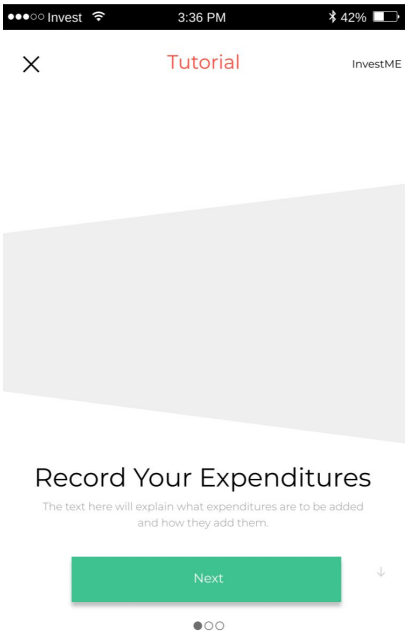
Allows users to login to InvestME. Here there will be 2 input fields and 3 clickable areas. The ‘Email Address’ and ‘Password’ sections are the input fields, and if you click on ‘Forgot your password?’ you will be linked to a page where you will be able to use your email to reset your password. If you click, ‘Log In’ you will be

2. Sign-up Page



Allows new users to create an account with InvestME. There are 4 input fields and 2 clickable buttons. Here they can choose to use their personal email addresses to sign-in. This will need to be confirmed upon verification sent to the email the user signed up with. Alternatively, they can choose to login with their Facebook or Google

3. Introductory Tutorial



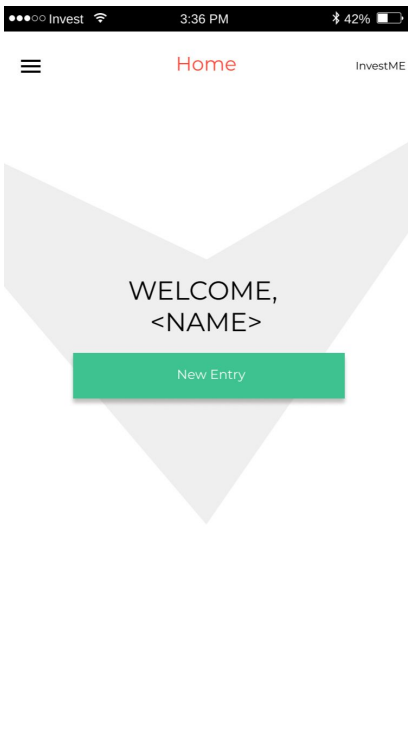
Allows new users to understand in 3 slides what InvestME is and how to use it. Using a slider we can move from page to page as the 3 circles at towards the bottom of the page indicate which page you are on. The next button will provide a way to move to the next page, however, users will also be able to swipe right to also do

logged into InvestMe. Lastly, if you click, ‘Sign Up’ you will be directed to a page allowing you to do so.

accounts.

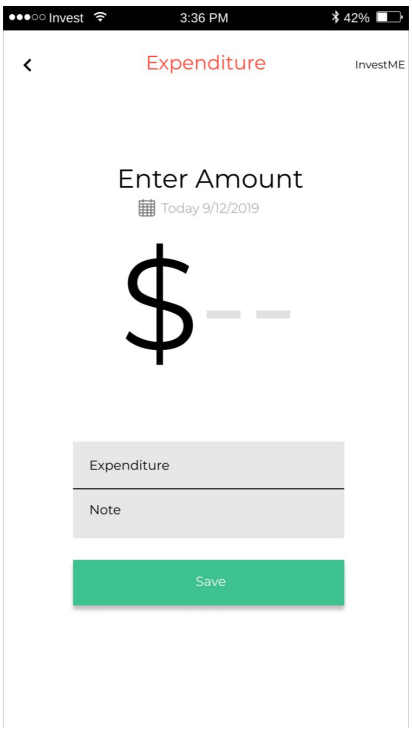
so. The cross in the top right allows the user to exit the tutorial.

4. Home Page



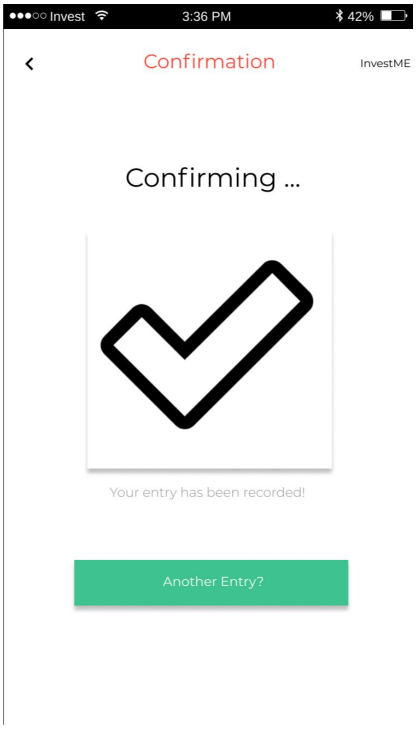
Allows users to create a new entry by clicking the, ‘New Entry’ button. The only other clickable area on this page is the hamburger icon. The hamburger when pressed opens the menu slider to easily navigate around InvestME.

5. New Entry Page



Allows users to create a new entry. There are 2 input fields and 3 clickable areas. The ‘Today <today’s date>’ section is a clickable area that defaults to the current date, but also can be clicked and changed if the user would like input a past or future entry. The calendar within the app will be able to do this as well, but for ease, the user is able to do this here as well. The ‘Expenditure’ input area allows the user to name the

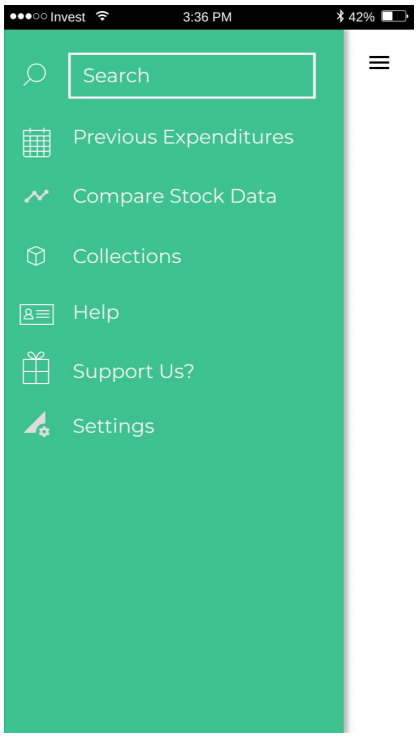
6. Confirmation Page



Allows users a way to wait for the entry to be saved if necessary. There is one clickable button. This page will also prompt the user if they want to add another entry by clicking the, ‘Another Entry’ button. The right angle bracket allows the user to go back to the previous page when clicked.

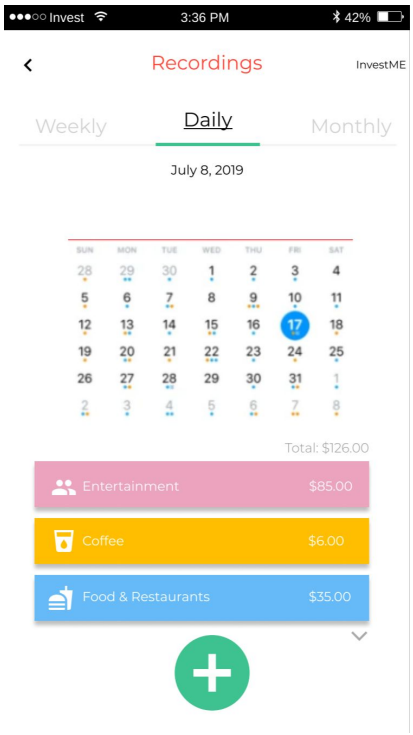
expenditure, while the note can be any relevant details or a more detailed description. The save button will save the expenditure. The right angle bracket allows the user to go back to the previous page when clicked.

7. Menu Slider Page



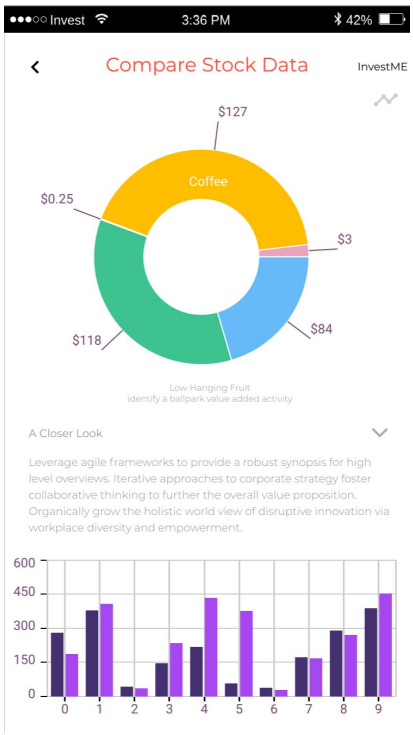
Allows users to easily navigate around InvestME. There are several clickable areas that will take the user to the appropriate page when clicked, e.g. clicking the, 'Previous Expenditures' will take the user to their recorded entries. If the user swipes left on the menu slider, the page will show whatever page the user was on prior to clicking the hamburger button.

8. Recordings



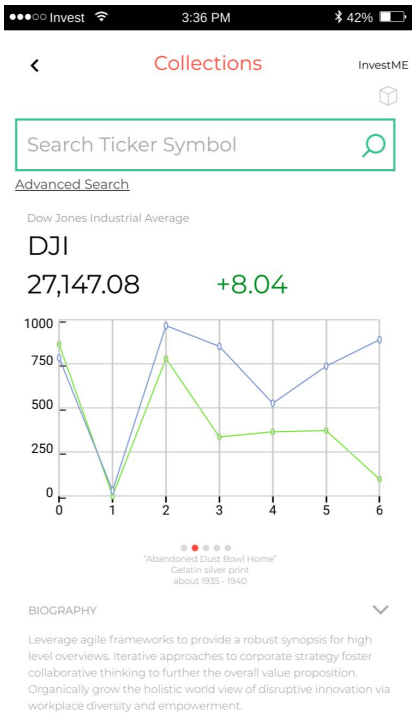
Allows users to view their recordings when they click, 'Previous Expenditures' in the menu slider. Here the users will be able to see all previous expenditures for any time frame. Additionally the user will have the ability to revise and add an entry here for ease. The right angle bracket allows the user to go back to the previous page when clicked.

9. Compare Stock Data Page



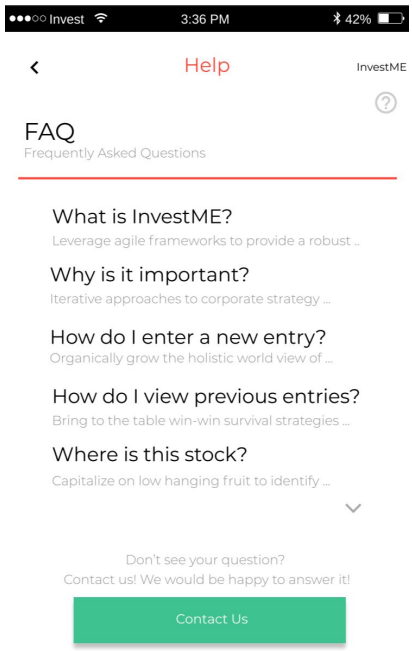
Allows users a way to compare their previous expenditures to different stocks. This page will contain diagrams and graphs to help explain this data. Additionally the paragraph of text will give more detailed information. The right angle bracket allows the user to go back to the previous page when clicked.

10. Collections Page



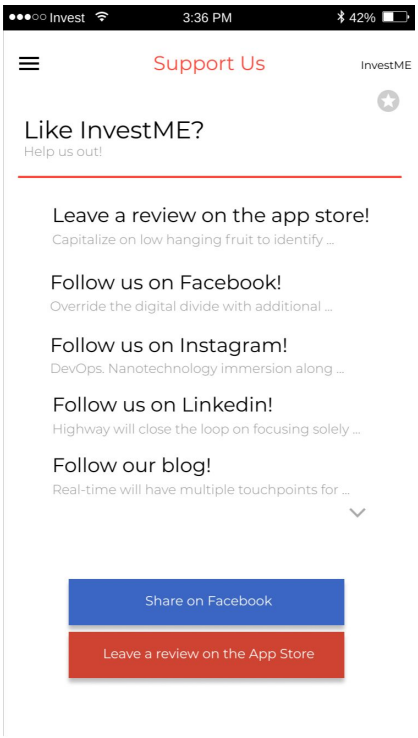
Allows users to easily search for a ticker symbol or global equity to find more information about that stock. There is 1 input area and 2 clickable areas. The box with the text, 'Search Ticker Symbol' will allow users to search for their desired symbol and gain more information. The advanced search clickable area will bring down a menu to filter the query. The right angle bracket allows the user to go back to the previous page when clicked.

11. Help/FAQ Page



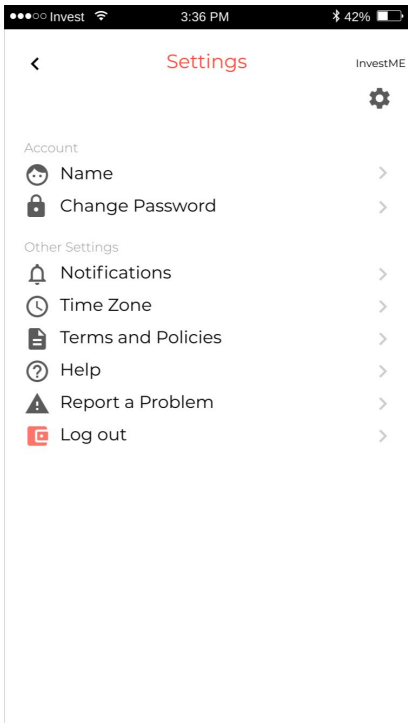
Allows users to view frequently asked questions and answers, and if necessary a way to contact the developers. The only clickable button will redirect them to email the developers at InvestME email account. The right angle bracket allows the user to go back to the previous page when clicked.

12. Support Us Page



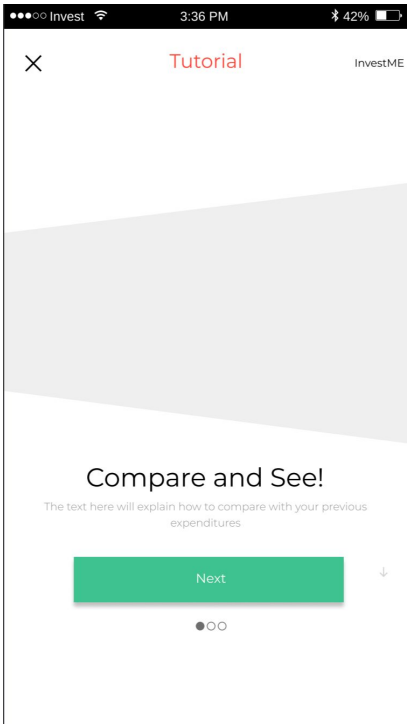
Allows users to help the developers gain a better following with this free-to-use app. Here all the information given is links to social media and other ways to assist the developers. The two buttons will allow the users to easily share InvestME on Facebook or review the InvestME on the app store. The right angle bracket allows the user to go back to the previous page when clicked.

13. Settings Page



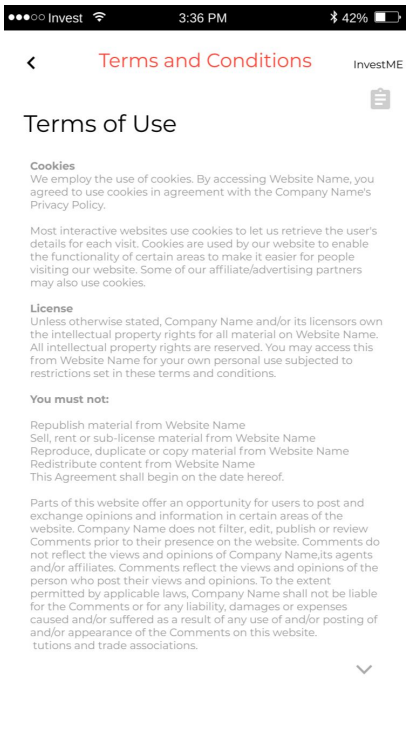
Allows users to access the settings within InvestME. The various settings will slide to the right to easily change and update. The right angle bracket allows the user to go back to the previous page when clicked.

14. Tutorial Page (opt.)



Allows new users to understand in 3 slides how to historical data. Using a slider we can move from page to page as the 3 circles at towards the bottom of the page indicate which page you are on. The next button will provide a way to move to the next page, however, users will also be able to swipe right to also do so. The cross in the top right allows the user to exit the tutorial.

15. Terms and Conditions



Allows users to view the terms and conditions of the app.The right angle bracket allows the user to go back to the previous page when clicked. The reverse chevron allows the user to view more. The right angle bracket allows the user to go back to the previous page when clicked.

4.2 User Effort Estimation

4.2.1 Recording a New Expenditure

This scenario represents recording a new expenditure (and saving it).

Navigation: Total of 3 taps, as follows

a. … Press “Log in with Google”
b. … Press “New Entry”
c. … Press “Save”

Data Entry: Total of 8 taps, and 0 swipes as follows

a. … Type “1” “5” to add a $15 expenditure
b. … Type “Coffee” under note

4.2.2 Comparing Expenditure

This scenario represents comparing expenditures with stock market data.

Navigation: Total of 3 taps, as follows

a. … Press “Log in with Google”
b. … Press the “Menu” icon
c. … Press “Compare Stock Data”

Data Entry: Total of 0 taps, and 0 swipes as follows

No data entry is required for this particular case.

4.2.3 Signing up for an InvestME account

This scenario represents signing up for an InvestME account.

Navigation: Total of 1 tap, as follows

a. … Press “Sign up”

Data Entry: Total of 5 taps, and 48 keystrokes as follows (Contingent on user’s input)

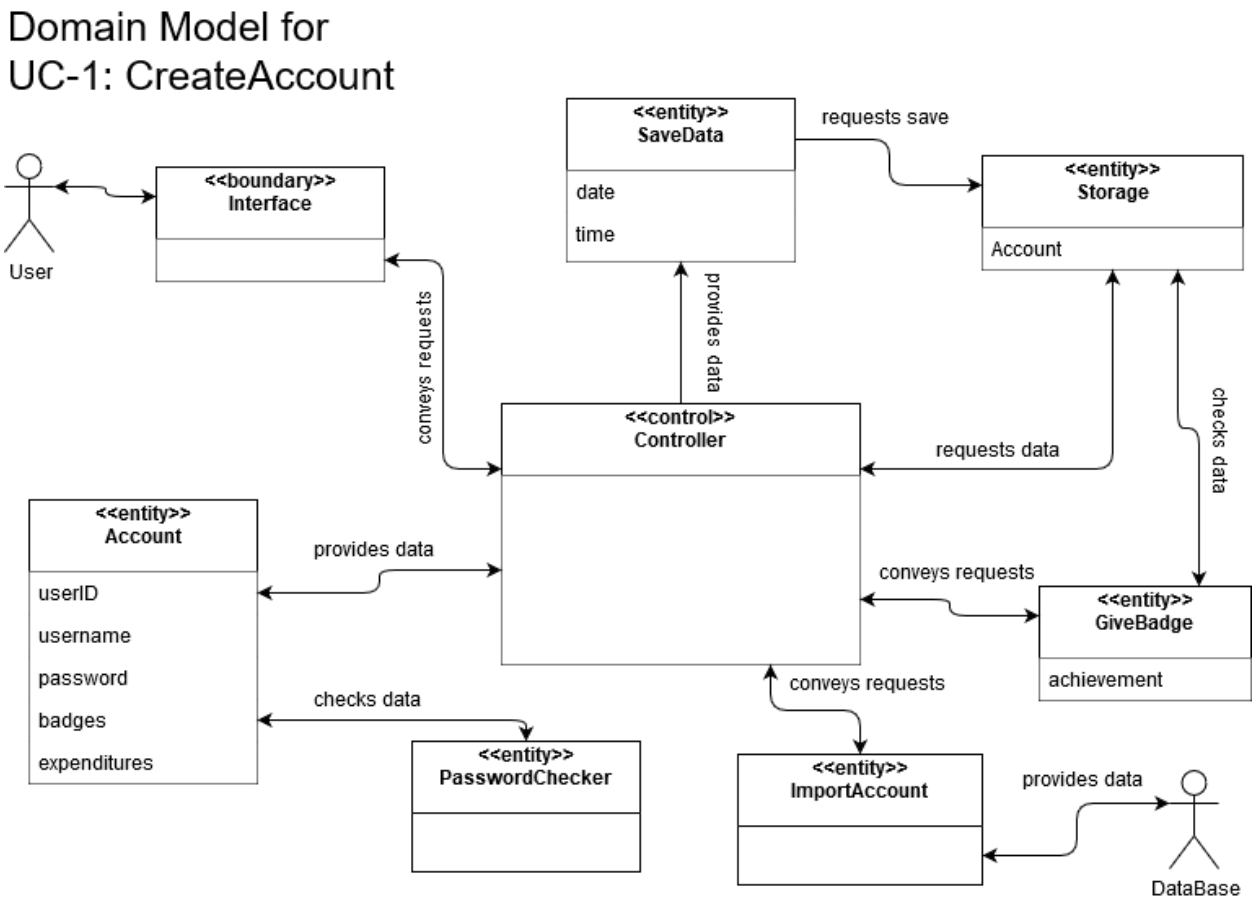
a. … Click cursor to “Name”
b. … Type “J” “o” “h” “n” “ “ “S” “m” “i” “t” “h”
c. … Click “Email Address”
d. … Type “J” “o” “h” “n” “S” “m” “i” “t” “h” “@” “a” “c” “m” “e” “.” “c” “o” “m”
e. … Click “Password”
f. … Type “m” “y” “p” “a” “s” “s” “w” “o” “r” “d”
g. … Click “Re-enter Password”
h. … Type “m” “y” “p” “a” “s” “s” “w” “o” “r” “d”
i. … Click “Submit”

Domain Analysis

5.1 Domain Model

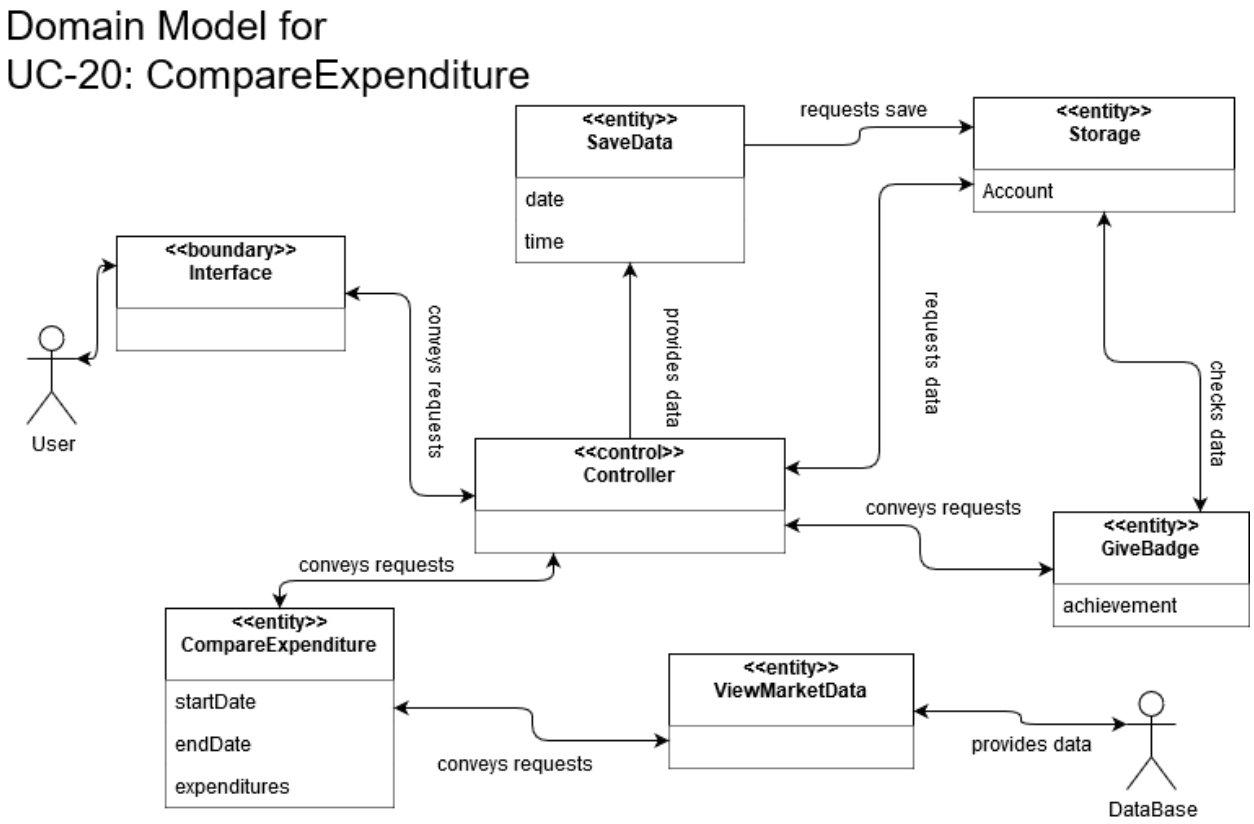
5.1.1 Concept Definitions

To analyze the domain model for each fully-dressed use case, a table of concept definitions and responsibilities are defined. The subsequent diagrams provide insight to how the concepts work together to complete each use case.



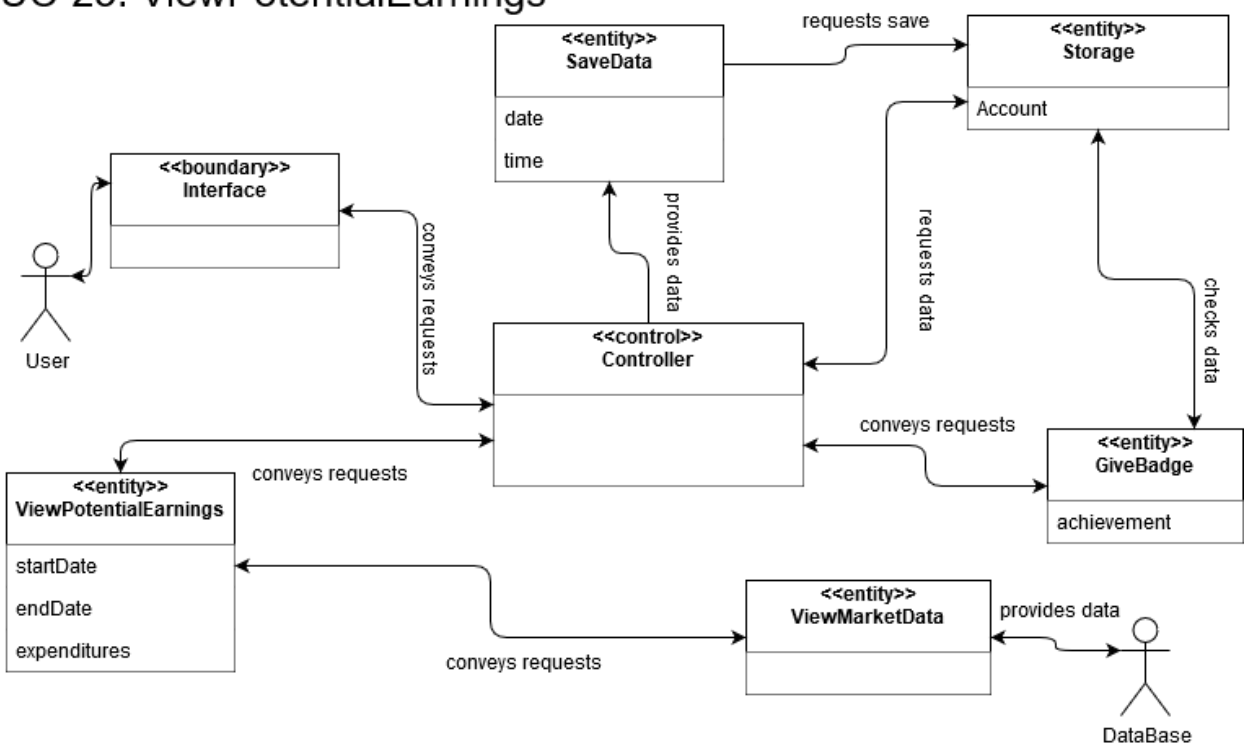
The diagram for UC-1 represents the user creating a new account with the InvestMe application. Initially, the user will interact with an interface and select the option to create a new account. When the user selects CreateAccount, the request is conveyed to the controller, who then delegates requests to other concepts. For example, for a new account, the controller will communicate the request to Account, who will then prompt the user to enter the necessary information to create a profile. When the user enters a password, Account will ask PasswordChecker to validate the password based on predefined criteria. If the user wishes to import a social media account, the controller will communicate to ImportAccount instead. ImportAccount will fetch the account being requested from an API DataBase outside of the InvestMe system. Once the account is verified from this request, ImportAccount will ask the

controller to save the account information to storage. The controller then delegates this request to SaveData, who puts a timestamp on the data and sends a request to Storage for archiving. InvestMe will give badges (user achievement awards) for fulfilling certain criteria such as creating a new account. Therefore, the Controller will request GiveBadge to check the Account in storage and see if it already has a badge or not. If the user is awarded a badge, GiveBadge will ask Controller to convey a message to the Interface informing the user has been awarded a badge.



The diagram for UC-20 represents the use case where the user wants to compare his or her expenditures with the various stocks available on the market in a specified timeframe. Initially, the user is interacting with the Interface and selects the option to compare their expenditures. Upon selecting, the controller delegates the request to CompareExpenditure who prompts the user to enter the startDate and endDate of the timeframe. After the user inputs the startDate and endDate, CompareExpenditure asks ViewMarketData to retrieve live stock market data from an API external to the system. Once retrieved, ViewMarketData will pass this information to CompareExpenditure, who will execute and display the comparison. After this, CompareExpenditure will convey the data back to the Controller, who then shares the rendered page with the Interface. The Controller will also ask GiveBadge to check if there is a user achievement that should be awarded to the user's account.

Domain Model for
UC-23: ViewPotentialEarnings



The diagram for UC-23 represents the use case where the user wants to view the potential earnings had they investing their expenditures into stocks. Initially, the user interacts with the Interface. Upon selecting the option to view potential earnings, the request is passed to the Controller who then delegates the task to ViewPotentialEarnings. ViewPotentialEarnings prompts the user to enter a startData and endDate for the time frame the user wishes to view any earnings. After ViewPotentialEarnings acquires the time frame, it will call ViewMarketData to retrieve stock market data from an API external to the system. ViewMarketData then conveys the information back to ViewPotentialEarnings, who then compares the expenditures with stocks that have increased since the expenditure was listed. Once this data is calculated, it is passed back to the Controller who displays the information to the Interface. The Controller will also call GiveBadge and ask to check if the user shall receive an achievement for completing the task.

Responsibility Description	Type	Concept Name
Coordinate the actions of concepts related to a use case, and delegate work to other concepts.	D	Controller
Container for storing and saving user data.	K	Storage
Graphical User Interface displaying the current context, what functions the actor can choose, or outcomes of various functions.	K	Interface
Generates a new account for a new user.	D	Account
Gives a badge when the user achieves a goal.	D	GiveBadge
Logs data from the user and stores the data into Storage.	D	SaveData
Checks the user password input with password criteria.	D	PasswordChecker
Imports the user's social media account.	D	ImportAccount
Calls ViewMarketData and compares logged expenditures with stocks in a specified timeframe.	D	CompareExpenditure
Share stock market information with other concepts.	D	ViewMarketData
Calls ViewMarketData and displays a list of stocks, along with their gains, equal to the total amount spent on expenditures.	D	ViewPotentialEarnings

5.1.2 Concept Associations

The concepts defined above need to operate in specific sequences. The following table provides corresponding association between each concept listed in the domain models.

Concept pair	Association description	Association name
Interface <-> Controller	Interface passes requests to Controller and receives back rendered pages.	conveys requests
Controller <-> Account	Controller passes requests to Account and receives back the account data.	provides data
Controller <-> PasswordChecker	Controller passes requests to PasswordChecker and receives back validation the entered password meets password criteria.	conveys requests

Controller <-> ImportAccount	Controller passes requests to ImportAccount and receives back the account imported.	conveys requests
ImportAccount <-> DataBase	DataBase retrieves the requested social media account and passes it to ImportAccount.	provides data
Controller <-> GiveBadge	Controller passes requests to GiveBadge and receives back the awarded badge.	conveys requests
GiveBadge <-> Storage	GiveBadge checks Storage to see if the user account already has a badge.	checks data
Controller <-> SaveData	Controller passes all acquired data to SaveData.	provides data
SaveData <-> Storage	SaveData provides a timestamp to the data given to it by the controller, then saves the data into Storage.	requests save
Controller <-> Storage	Controller passes a request to Storage to retrieve Account data.	requests data
Controller <-> CompareExpenditure	Controller passes a request to CompareExpenditure and retrieves a rendered display of compared stocks and expenditures.	conveys requests
CompareExpenditure <-> ViewMarketData	CompareExpenditure passes a request to ViewMarketData and receives stock information within the specified timeframe.	conveys requests
ViewMarketData <-> DataBase	DataBase retrieves the requested stock market data and passes it to ViewMarketData.	provides data
Controller <-> ViewPotentialEarnings	Controller passes a request to ViewPotentialEarnings and receives a rendered display of stocks that have increased in value.	Conveys requests

5.1.3 Attribute Definitions

Attribute	Attribute Definition	From Concept
userID	A unique ID attached to each	Account

	account.	
userName	The name associated with the account.	Account
password	The key used with the username to login to each account. The password will have specific restrictions to ensure account security.	Account
badges	A token for users to put in their account demonstrating their completion of a specific task within the application.	Account
expenditures	User input that contains a name and amount, representing money spent.	Account
timestamp	Keeps the date and time the data was saved.	SaveData
logData	Saves and updates account information..	SaveData
Account	An instance of the Account class that will be saved. Multiple Account classes will be saved within the database	Storage
achievement	The information(name and date earned) associated with the particular badge.	GiveBadge
startDate	The date to begin checking expenditure timeStamps from.	CompareExpenditure
endDate	The date to stop checking expenditure timestamps on.	CompareExpenditure
expenditures	The data inputted from the user and being compared to stock market data (taken from the API).	CompareExpenditure
startDate	The date to begin checking expenditure timeStamps	ViewPotentionalEarnings

	from.	
endDate	The date to stop checking expenditure timestamps on.	ViewPotentionalEarnings
expenditures	The expenditures and subsequent earnings(being taken from the stock API) being compared.	ViewPotentionalEarnings

5.1.4 Traceability Matrix

Use Case	Priority Weight	Controller	Storage	Interface	Account	GiveBadge	SaveData	PasswordChecker	ImportAccount	Compare Expenditure	ViewMarketData	ViewPotentialEarnings
UC-1	10	X	X	X	X	X	X	X	X	X	X	
UC-20	10	X	X	X		X	X			X	X	
UC-23	10	X	X	X		X	X				X	X
MAX PW		10	10	10	10	10	10	10	10	10	10	10
TOTAL PW		30	30	30	10	30	30	10	10	20	30	10

For a link to the full-sized matrix click here. This matrix shows the domain concepts correspond with the three fully-dressed use cases. In addition, the priority weight of the use cases can be seen at the bottom.

5.2 System Operation Contracts

UC-1 Register/Create An Account

Preconditions

- (join) If a new user wants to use the InvestME app, they must sign up for an account first. Users will have the option of registering a social media account or a public email address.
- User would like to manually enter information to create an account.

Postconditions

- A new account is successfully created.
- A badge is given to the user for completing the “Create Account” achievement.

UC-2 Account Modification

Preconditions

- Users must be logged into their InvestME account on their mobile device or in the web browser to make modifications.

Postconditions

- Account successfully modified.
- Database has been updated.

UC-3 Account Password Reset

Preconditions

- The user is unable to remember account password.

Postconditions

- The user is able to successfully change account password.

UC-4 Account Deletion

Preconditions

- Users must be logged into their InvestME account.
- Users must confirm the deletion of their InvestME account.

Postconditions

- User's InvestME account is securely deleted from the database.

UC-5 Terms and Conditions

Preconditions

- User is signing up for an InvestME account.
- User is signed into existing InvestME account.

Postconditions

- User agrees to the terms and conditions to complete the InvestME account sign-up. Account in database updated to show the user accepted the terms and conditions.
- The terms and conditions are viewed by the user.

UC-6 Achievements and Badges

Preconditions

- User must be logged into their InvestME account.
- User must have an existing InvestME account.
- Predetermined milestone is met.

Postconditions

- User's account receives achievement awards and a badge for the completion of a milestone.
- Database updated to show completion of milestones.

UC-7 Tutorials

Preconditions

- User must be logged into their InvestME account.
- Tutorials link is available.

Postconditions

- Tutorials successfully retrieved by the user.

UC-8 Data Input

Preconditions

- User must be logged into their InvestME account.
- Database is available.

Postconditions

- Database has been updated.

UC-9 Cloud Storage

Preconditions

- User must be logged into their InvestME account.
- User must link cloud service account to the InvestME app

Postconditions

- Database is updated to reflect additional save location.

UC-10 Track Expenditures

Preconditions

- User must be logged into their InvestME account.
- InvestME app database available

Postconditions

- Query user expenditure data from database.

UC-11 Recommendations

Preconditions

- User must be logged into their InvestME account.
- Alpha Vantage database is accepting inquiries
- User InvestME data is available.

Postconditions

- InvestME app provides recommendations based on user's expenditure dollar amounts.

UC-12 View Data

Preconditions

- User must be logged into their InvestME account.
- Alpha Vantage database is accepting inquiries.
- Ticker symbol or registered company name known.

Postconditions

- Query stock market data.

UC-13 Search Data

Preconditions

- User must be logged into their InvestME account.
- Alpha Vantage database is accepting inquiries.
- Ticker symbol or registered company name known.

Postconditions

- Query stock market data.

UC-14 Compare Expenditure

Preconditions

- User must be logged into their InvestME account.
- InvestME Database is available.
- Alpha Vantage database is accepting inquiries.

Postconditions

- Amount of expenditures from date range calculated.
- Query stock market historical data.
- Provide analysis on how much could have been saved.

UC-15 Save Data

Preconditions

- User must be logged into their InvestME account.
- Alpha Vantage database is accepting inquiries.

Postconditions

- Query stock market data.
- User account database has been updated.

UC-16 Update Data

Preconditions

- User must be logged into their InvestME account.
- Alpha Vantage database is accepting inquiries.
- InvestME Database is available.

Postconditions

- The InvestME app updates stock information of companies being followed.
- User database updated.

UC-17 User Requests

Preconditions

- User must be logged into their InvestME account.
- User requests information on how to purchase shares

Postconditions

- The user receives information on how to purchase shares.

UC-18 Daily Notifications

Preconditions

- User must be logged into their InvestME account.
- Notifications are enabled by the user.

Postconditions

- User receive daily notifications to input expenditures.

UC-19 Geo-Fencing Notifications

Preconditions

- User must be logged into their InvestME account.
- Geo-Fencing notifications are enabled by the user.

Postconditions

- User receive notifications based on their current location.

5.3 Mathematical Model and APIs

Alpha Vantage APIs

Since InvestME will be targeting those that want to learn more about why they should consider investing, it is important to understand how we are going to provide users with real-time and over 20 years of historical stock data. Fortunately for us, Alpha Vantage produces APIs (Application Program Interface) that will be able to assist us to do just that.

The APIs they offer attack four specific collections to do just that.

1. Stock Time Series Data
2. Physical and Digital/Crypto Currencies (e.g., Bitcoin, Litecoin, XRP)
3. Technical Indicators
4. Sector Performance

5.3.1 Stock Time Series Data

We can use an API to show us the daily stock time series (open, high, low, close) of a global equity considering a specific time frame.

This suite of APIs provide realtime and historical global equity data in 4 different temporal resolutions: (1) daily, (2) weekly, (3) monthly, and (4) intraday. Daily, weekly, and monthly time series contain 20+ years of historical data.

Times Series Intraday

This API returns intraday time series (timestamp, open, high, low, close, volume) of the equity specified.

Meta Data:

1. Information: Daily Prices (open, high, low, close) and Volumes
2. Symbol: MSFT
3. Last Refreshed: 2019-09-20 14:41:22
4. Output Size: Compact
5. Time Zone: US/Eastern

Time Series (Daily):

2019-09-20:

1. open: 141.1500
2. high: 141.6500
3. low: 138.2500
4. close: 139.3825

5. volume: 19102467

2019-09-19:

1. open: 140.3000
2. high: 142.3700
3. low: 140.0798
4. close: 141.0700
5. volume: 34823372

2019-09-18:

1. open: 137.3600
2. high: 138.6700
3. low: 136.5300
4. close: 138.5200
5. volume: 23982100

5.3.2 Physical and Digital/Crypto Currencies

We will use these APIs to give us the ability to show users how much they would have saved at a specific time had they invested in this global equity in different ways with a variety of approaches.

Meta Data:

1. From_Currency Code: BTC
2. From_Currency Name: Bitcoin
3. To_Currency Code: CNY
4. To_Currency Name: Chinese Yuan
5. Exchange Rate: 70795.43856000
6. Last Refreshed: 2019-09-18 08:36:09
7. Time Zone: UTC
8. Bid Price: 70789.80744000
9. Ask Price: 70798.64616000

5.3.3 Technical Indicators

Technical indicator values are updated real time: the latest data point is derived from the current trading day of a given equity or currency exchange pair.

SMA

This API returns the simple moving average (SMA) values. See also: Investopedia article and mathematical reference.

Mathematical Model:

$$SMA = \frac{\sum_{1}^{n} price}{n}$$

Meta Data:

1: Symbol: MSFT

2: Indicator: Simple Moving Average (SMA)

3: Last Refreshed: 2019-09-20

4: Interval: weekly

5: Time Period: 10

6: Series Type: open

7: Time Zone: US/Eastern

Technical Analysis: SMA:

2019-09-20:

SMA: 137.3590

2019-09-13:

SMA: 137.4160

2019-09-06:

SMA: 137.1200

2019-08-30:

SMA: 137.1590

2019-08-23:

SMA: 136.9230

2019-08-16:

SMA: 136.3780

5.3.4 Sector Performances

This API returns the realtime and historical sector performances calculated from S&P500 incumbents.

Meta Data:

Information: US Sector Performance (realtime & historical),

Last Refreshed: 04:20 PM ET 09/20/2019

Rank A: Real-Time Performance:

Health Care: 0.60%

Utilities: 0.37%

Energy: 0.08%

Materials: -0.16%

Consumer Staples: -0.22%

Real Estate: -0.33%

Communication Services: -0.46%

Financials: -0.64%

Industrials: -0.66%

Information Technology: -1.12%

Consumer Discretionary: -1.17%

Rank B: 1 Day Performance:

Health Care: 0.60%

Utilities: 0.37%

Energy: 0.08%

Materials: -0.16%

Consumer Staples: -0.22%

Real Estate: -0.33%

Communication Services: -0.46%

Financials: -0.64%

Industrials: -0.66%

Information Technology: -1.12%

Consumer Discretionary: -1.17%

Project Size Estimation

7.1 Use Case Points

Complex projects require more time and effort to design and implement. Thus, there is a need to use a method to estimate the time it takes to complete a project. Use Case Points (UCP) can be used to estimate the amount of person-hours it takes to complete a software project. There are several factors used to estimate the time:

- Functional Requirements - often represented as use cases. The complexity of use cases is determined by the number and complexity of actors, as well as how many steps it takes to complete each use case.
- Nonfunctional requirements - known as FURPS+, and includes properties such as security, usability, and performance. These are also called the technical complexity factors.
- Environmental factors - this includes the experience and knowledge of the development team, as well as the level of sophistication in the tools they use for development.

Understanding this project completion time is important for project scheduling, cost, and resource allocation. The following formula is used to calculate the time, and uses three variables:

$$UCP = UUCP \times TCF \times ECF$$

1. Unadjusted Use Case Points (UUCP) - measures the complexity of the functional requirements.
2. Technical Complexity Factor (TCF) - measures the complexity of the nonfunctional requirements.
3. Environment Complexity Factor (ECF) - assesses the experience of the development team, and their development environment.

For this project, the assumption is every student is working under the same “environmental factors”, so we will use ECF = 1 for the UCP formula. Only UUCP and TCF will be calculated based on the detailed use cases. The actors in each use case are classified and have associated weights. Below is a table showing this rubric:

Actor type	Description of how to recognize the actor type	Weight
Simple	The actor is another system which interacts with our system through a defined application programming interface (API).	1
Average	The actor is a person interacting through a text-based user interface, or another system interacting through a protocol, such as a network communication protocol.	2
Complex	The actor is a person interacting via a graphical user interface.	3

7.1.1 Unadjusted Use Case Points (UUCP)

Calculating UUCPs involves the sum of the following two components:

1. Unadjusted Actor Weight (UAW) - the combined complexity of each actor in all the use cases.
2. Unadjusted Use Case Weight (UUCW) - the total number of steps contained in all the use case scenarios.

To calculate UAW, we will use the table below:

Actor name	Description of relevant characteristics	Complexity	Weight
Database	Database is another system acting through a protocol.	Simple	1
System	System is performing automated functions acting through protocol.	Average	2
User	User is interacting with the system via a graphical user interface.	Complex	3

UAW = (3 x Complex) + (2 x Average) + (1 x Simple) = (3 x 1) + (2 x 1) + (1 x 1) = 6

UUCW is calculated in a similar way by analyzing our use cases. We can classify them into three categories: Simple, Average, and Complex. We find out which category a use case belongs to by considering how many steps it takes to complete a use case. This includes the main success scenario, as well as the alternative scenarios. UUCW is calculated by tallying the amount of use cases within each of the three categories. After this, the totals are multiplied by the weight factor, and then by summing the products.

Use case category	Description of how to recognize the use-case category	Weight
Simple	Simple user interface. Up to one participating actor (plus initiating actor). Number of steps for the success scenario: ≤ 3. If presently available, its domain model includes ≤ 3 concepts.	5
Average	Moderate interface design. Two or more participating actors. Number of steps for the success scenario: 4 to 7. If presently available, its domain model includes between 5 and 10 concepts.	10
Complex	Complex user interface or processing. Three or more participating actors. Number of steps for the success scenario: ≥ 7. If available, its domain model includes ≥ 10 concepts.	15

Use case	Description	Category	Weight
CreateAccount (UC-1)	Average user interface. 7 steps for the main success scenario. 2 participating actors (System, DataBase). 8 concepts in domain model.	Average	10
CompareExpenditure (UC-20)	Average user interface. 7 steps for the main success scenario. 2 participating actors (System, DataBase). 7 concepts in domain model.	Average	10
ViewPotentialEarnings (UC-23)	Average user interface. 7 steps for the main success scenario. 2 participating actors (System, DataBase). 7 concepts in domain model.	Average	10

UUCW = (0 x Complex) + (3 x Average) + (0 x Simple) = (0 x 15) + (3 x 10) + (0 x 5) = 30

UUCP = UAW + UUCW = 6 + 30 = 36

7.1.2 Technical Complexity Factor (TCF)

There are 13 standard technical factors used to estimate the impact on productivity for the nonfunctional requirements in a project. Each factor has a weight according to its impact. It is up to the development team to assess the complexity of each technical factor according to the table below:

Technical factor	Description	Weight
T1	Distributed system	2
T2	Performance objectives	1
T3	End-user efficiency	1
T4	Complex internal processing	1
T5	Reusable design or code	1
T6	Easy to install	0.5
T7	Easy to use	0.5
T8	Portable	2

T9	Easy to change	1
T10	Concurrent use	1
T11	Special security features	1
T12	Provides direct access for third parties	1
T13	Special user training facilities are required	1

Based on the assessment, the team assigns a perceived complexity value between 0 and 5 (0 meaning irrelevant, 3 meaning the effort is average, and 5 corresponds to major effort). This value is based on how much effort will be needed to satisfy a nonfunctional requirement. Each factor's weight is multiplied by its perceived complexity factor. These calculated factors are then summed to produce the TCF. Two constants are used to produce the formula for the TCF, constant 1 (0.6) and constant 2 (0.01). The constants limit the impact the TCF has on the UCP equation.

Technical factor	Description	Weight	Perceived complexity	Calculated Factor
T1	Distributed System (running on multiple devices)	2	3	2 x 3 = 6
T2	Users expect good performance, but nothing exceptional	1	3	1 x 3 = 3
T3	End-user expects efficiency, but there are no exceptional demands	1	3	1 x 3 = 3
T4	Internal processing requires interaction with other subsystems	1	4	1 x 4 = 4
T5	No requirement for reusability	1	1	1 x 1 = 1
T6	Ease of install is relatively simple (will be installed via smartphone app stores)	0.5	2	0.5 x 2 = 1
T7	Ease of use is very important	0.5	5	0.5 x 5 = 2.5
T8	Portability is moderately important (should run on 2 major platforms, Apple IOS and Android)	2	3	2 x 3 = 6
T9	Relatively simple to change or add new features	1	2	1 x 2 = 2

T10	Concurrent use is required	1	4	1 x 4 = 4
T11	Security is an moderate concern	1	3	1 x 3 = 3
T12	No direct access for third parties	1	0	1 x 0 = 0
T13	Some training is needed (for those unfamiliar with programming language and IDE)	1	2	1 x 2 = 2
			Technical Factor Total:	37.5

Constant-1 (C1) = 0.6, Constant-2 (C2) = 0.01

TCF = 0.6 + (0.01 x 37.5) = 0.975

Thus, the results decrease the UCP by 2.5%

7.1.3 Environment Complexity Factor (ECF)

The ECFs measure the experience level of the people on the project, as well as the stability of the project. Team members with more experience will reduce the UCP count, and members with less experience will increase the UCP count. Other external factors are considered such as: Available budget, company's market position, and the state of the economy. However, per the instructions for this project, we are to assume all students will work under the same "environmental factors." Thus, we are not to calculate the standard ECF for our project, and for this project, the ECF shall equal 1.

7.1.4 Calculated the Use Case Points (UCP)

Now that we have our totals for UUCP, TCF, and ECF, we can plug in each value in the formula:

UCP = UUCP x TCF x ECF = 36 x 0.975 x 1 = 35.1 use case points

Plan of Work

The plan of work acts as our guideline to ensure that we remain on the same page throughout the process. We are implementing Agile/SCRUM principles. The main document provides a detailed breakdown of each task and its corresponding subtasks. Each task has a set number of story points to assist in properly delegating and cataloging the work completed. Supplementing this document is a burndown page with a corresponding chart to show progress. This creates a visual representation for the development team as we continuously progress in the project's development.

In addition to these two documents, the team will be using Trello, a project management web application. This will allow us to keep track of any deadlines in a roadmap and ensure that any complications within a specific task are addressed properly and efficiently. Finally, we will meet every morning for a minimum of 10 minutes to confirm the daily agenda. Accompanying this, we will have a 30 minute to an hour long meeting every Saturday (day subject to change) to debrief and prepare for the following week. All of these tools will be used to complete the project in a timely manner.

7.1 Detailed Plan of Work

For a better view of this table, please click here.

		START DATE	END DATE	PROJECT MEMBERS	PROJECT LEAD		CURRENT SPRINT PROGRESS	OVERALL PROGRESS		
		8/22/19	12/9/19	Nick	Kristyna		0%	0%		
				John						
				Jeremiah						
				Kristyna						

	PRIORITY	TASK NAME	DETAILS	USER STORIES	RESPONSIBLE	STORY POINTS	START	FINISH	STATUS	COMMENTS
	HIGH	**Design Log-in page**	Create LoginPage	ST-1	Nick	Total: 4 points	9/24/19	10/7/19	NOT STARTED	
	HIGH	Title	*Show application name*		John	1 point			NOT STARTED	
	HIGH	Form Fields	*Create two field: address, password*		Jeremiah	1 point			NOT STARTED	
	HIGH	Login	*Create login button*		Kristyna	1 point			NOT STARTED	
	HIGH	Navigate	*Create link to Sign-up Page*		Nick	1 point			NOT STARTED	
	LOW	**Give Badges**	Attach badge to users account	ST-6	John	Total: 4 points	9/24/19	10/7/19	NOT STARTED	
	LOW	Create badge	Develop method for badge creation		Jeremiah	2 points			NOT STARTED	
	LOW	Add badge	Develop method for adding badges to account		Kristyna	2 points			NOT STARTED	
	HIGH	**Develop Login Methods**	Verify user email and password for login page.	ST-1	Nick	Total: 4 points	9/24/19	10/7/19	NOT STARTED	

	HIGH	Validate: userName	*Validate input: userName*		John	2 points			NOT STARTED
	HIGH	Validate: password	*Validate input: password*		Jeremiah	2 points			NOT STARTED
	HIGH	**Design Sign-Up page**	Create Sign-Up page	ST-1	Kristyna	Total: 4 points	9/24/19	10/7/19	NOT STARTED
	HIGH	Title	*Show application name*		Nick	1 point			NOT STARTED
	HIGH	Form Fields	*Create input fields: name, email address, password, retype password*		John	3 points			NOT STARTED
	LOW	**Login from 3rd Party**	Login with Facebook or Gmail	ST-1	Jeremiah	Total: 6 points			NOT STARTED
	LOW	Facebook	*Button: login with Facebook*		Kristyna	3 points			NOT STARTED
	LOW	Google	*Button: login with Gmail*		Nick	3 points			NOT STARTED
	LOW	**Implement terms and conditions**	Accept terms and conditions before creation of new account	ST-5	John	Total: 4 points	9/24/19	10/7/19	NOT STARTED

	LOW	Prompt	*Create prompt prior to account creation*		Jeremiah	1 point			NOT STARTED
	LOW	Describe terms	*Describe terms and conditions*		Kristyna	1 point			NOT STARTED
	LOW	Accept	*Accept or decline terms*		Nick	2 points			NOT STARTED
	LOW	**Design tutorial page**	First time users directed to tutorial page for inputting expenditures and application goals.	ST-7	John	Total: 4 points	9/24/19	10/7/19	NOT STARTED
	LOW	Navigation	*Direct user to tutorial*		Jeremiah	1 point			NOT STARTED
	LOW	Explanation	*Explain the application and inputting data*		Kristyna	2 points			NOT STARTED
	LOW	Exit	*Exit the tutorial and navigate to home page*		Nick	1 point			NOT STARTED
	MEDIUM	**Design the confirmation page**	Confirmation of successful input	ST-8	John	Total: 5 points	9/24/19	10/7/19	NOT STARTED
	MEDIUM	Alert	*Alerts that an input is being save*		Jeremiah	1 point			NOT STARTED

	MEDIUM	Confirmation	*Confirms save to the database*		Kristyna	3 points			NOT STARTED
	MEDIUM	Navigation	*Button: navigate to home page*		Nick	1 point			NOT STARTED
	HIGH	**Design the Home Page**	Main screen of the application. Easy access to inputting expenditures.	ST-8	John	Total: 2 points	9/24/19	10/7/19	NOT STARTED
	HIGH	Title	*Username as the title with welcome message*		Jeremiah	1 point			NOT STARTED
	HIGH	Navigate	*Button: navigate to new entry page*		Kristyna	1 point			NOT STARTED
	MEDIUM	**Design the Menu Page**	Contains user settings, expenditure settings, contact information	ST-11, ST-18, ST-16, ST-14, ST-12	Nick	Total: 7 points	9/24/19	10/7/19	NOT STARTED
	MEDIUM	Navigate	*Icon on all main pages pulls up menu*		John	2 points			NOT STARTED
	MEDIUM	Search	*Input: Search box*		Jeremiah	2 points			NOT STARTED
	MEDIUM	Navigation	*Links: previous expenditurs, compare*		Kristyna	3 points			NOT STARTED

			Stock Data, Collections, Help , Support Us?						
	MEDIUM	**Develop methods dealing with account security**	Security related methods	ST-2, ST-3, ST-4	Nick	Total: 9 points	9/24/19	10/7/19	NOT STARTED
	MEDIUM	edit password	*edit user's password.*		John	3 points			NOT STARTED
	MEDIUM	reset password	*Reset user's password via automated email*		Jeremiah	4 points			NOT STARTED
	MEDIUM	delete account	*Removes the user account and associated data from the database*		Kristyna	2 points			NOT STARTED
	HIGH	**Design New Entry page**	Create New Entry page	ST-8	Nick	Total: 4 points	9/24/19	10/7/19	NOT STARTED
	HIGH	Input data	*Input name and amount of expenditure*		John	2 points			NOT STARTED
	HIGH	Save / Navigate	*Button: save user input and navigate to confirmation page*		Jeremiah	2 points			NOT STARTED

	HIGH	**Develop method for saving entries to the Database**	Create methods for cloud storage	ST-9	Kristyna	Total: 8 points	9/24/19	10/7/19	NOT STARTED
	HIGH	Layout	*Create Scaffold for loading screen*		Nick	2 point	10/8/19	10/21/19	NOT STARTED
	HIGH	Save	*Store in database*		John	3 points	10/8/19	10/21/19	NOT STARTED
	HIGH	Load	*Retrieve from database*		Jeremiah	3 points	10/8/19	10/21/19	NOT STARTED
	HIGH	**Develop methods for adding expenditures**	Create methods to intake expenditures	ST-8	Kristyna	Total: 9 points	10/8/19	10/21/19	NOT STARTED
	HIGH	Layout	*Create Scaffold*		Nick	1 point	10/8/19	10/21/19	NOT STARTED
	HIGH	Add expenditure	*Add expenditure item*		John	2 points	10/22/19	11/5/19	NOT STARTED
	HIGH	Delete expenditure	*Remove expediture item*		Jeremiah	2 points	10/22/19	11/5/19	NOT STARTED
	HIGH	Edit expenditure	*Edit expenditure item*		Kristyna	2 points	10/22/19	11/5/19	NOT STARTED
	HIGH	Load expenditure	*Retrieve from database*		Nick	2 points	10/22/19	11/5/19	NOT STARTED
	HIGH	**Develop introdu**	Tutorial explains/ encourage	ST-10	John	Total: 6 points	10/22/19	11/5/19	NOT STARTED

		ctory tutorial	s how to track previous expenditures						
	HIGH	Layout	*Create appropriate scaffolds for tutorial*		Jeremiah	2 point	10/22/19	11/5/19	**NOT STARTED**
	HIGH	Design Slides	*Create tutorial slideshow*		Kristyna	3 points	9/17/19	9/22/19	**NOT STARTED**
	HIGH	Implement slides	*Write tutorial slide information*		Nick	1 point	9/17/19	9/22/19	**NOT STARTED**
	MEDIUM	**Develop Records Page**	Displays recordings of previous expenditures and allows user revision of inputs	ST-11	John	Total: 9 points	9/17/19	9/22/19	**NOT STARTED**
	MEDIUM	Layout	*Create page layout*		Jeremiah	2 points			**NOT STARTED**
	MEDIUM	Display	*Display previous expenditures*		Kristyna	2 points			**NOT STARTED**
	MEDIUM	Revise	*Revise previous expenditures*		Nick	3 points			**NOT STARTED**
	MEDIUM	Add	*Add expenditure*		John	2 points			**NOT STARTED**

	HIGH	**Basic setup database**	Set up the database for the application	ST-1, ST-2, ST-3, ST-4, ST- 8, ST-9	Jeremiah	Total: 11 points	9/17/19	9/22/19	NOT STARTED
	HIGH	Setup	*Add dependencies*		Kristyna	2 point	9/17/19	9/22/19	NOT STARTED
	HIGH	Implement	*Create database client*		Nick	5 points	9/17/19	9/22/19	NOT STARTED
	HIGH	Create model(s)	*Create model Class*		John	1 point	9/17/19	9/22/19	NOT STARTED
	HIGH	Setup	*Add dependencies*		Jeremiah	2 point	9/17/19	9/22/19	NOT STARTED
	HIGH	**CRUD operation**	*Appropriate operations needed for database*	ST-8, ST-9, ST-4, ST-3, ST-2, ST-2	Kristyna	Total: 4 points			NOT STARTED
	HIGH	Create	*Create operation*		Nick	1 point			NOT STARTED
	HIGH	Read	*Read operation*		John	1 point			NOT STARTED
	HIGH	Update	*Update operation*		Jeremiah	1 point			NOT STARTED
	HIGH	Delete	*Delete operation*		Kristyna	1 point			NOT STARTED
	MEDIUM	**BLoC operation**	*Appropriate operations needed for database*	ST-8, ST-9	Nick	Total: 7 points			NOT STARTED
	MEDIUM	Received call	*Get clients operation*		John	1 point			NOT STARTED

	MEDIUM	Open stream	*Stream controller*		Jeremiah	3 point			NOT STARTED
	MEDIUM	Close stream	*Close stream*		Kristyna	1 points			NOT STARTED
	MEDIUM	Perform test	*Test and interact*		Nick	2 points			NOT STARTED
	MEDIUM	**Set up Alpha Vantage APIs**	Integrate stock market API	ST-12, ST-14, ST-16, ST-21	John	Total: 8 points			NOT STARTED
	MEDIUM	Get intraday time series	*Implement Stock Time Series Data API*		Jeremiah	2 points			NOT STARTED
	MEDIUM	Get exchange rate	*Physical and Digital/Crypto Currencies API*		Kristyna	2 points			NOT STARTED
	MEDIUM	Get SMA values	*Technical Indicators API*		Nick	2 points			NOT STARTED
	MEDIUM	Get S&P500 Performance	*Sector Performances API*		John	2 points			NOT STARTED
	LOW	**Design Historical Data Tutorial page**	This tutorial will show the user how to view and analyze historical stock market data and apply it to the various entries.	ST-13	Jeremiah	Total: 6points			NOT STARTED

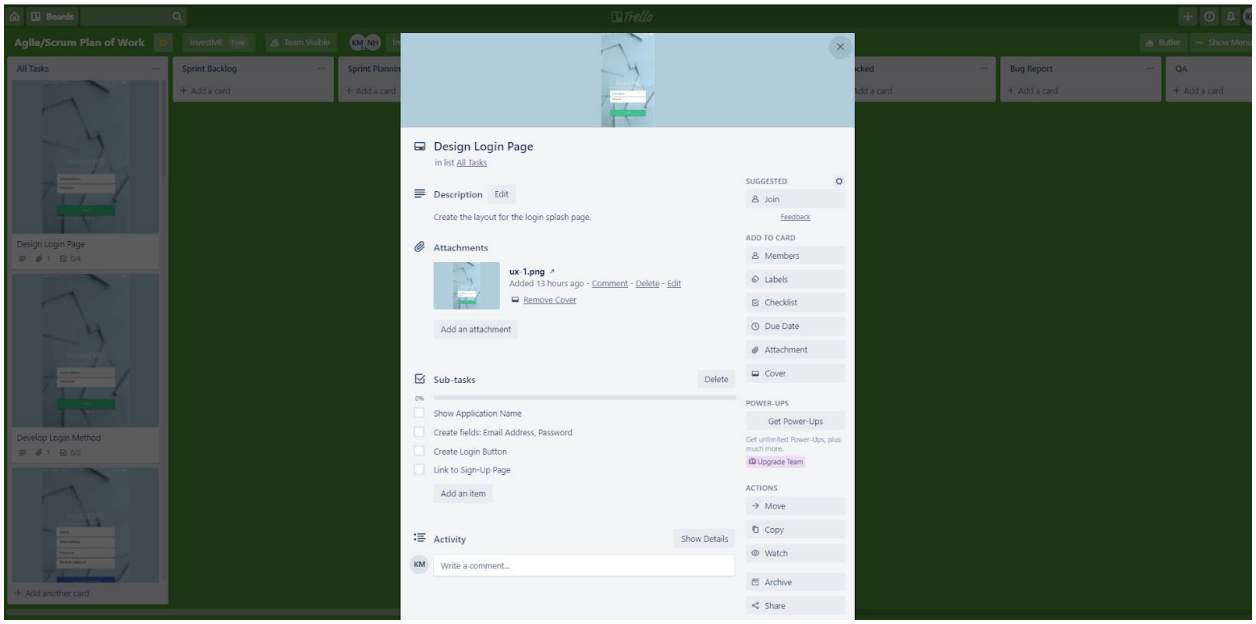
	LOW	Layout	*Create appropriate scaffolds for tutorial*		Kristyna	2 points			NOT STARTED
	LOW	Design Slides	*Create tutorial slideshow*		Nick	2 points			NOT STARTED
	LOW	Implement slides	*Write tutorial slide information*		John	2 points			NOT STARTED
	MEDIUM	**Develop method to compare stock data**	Compare user expenditures with stock data	ST-14	Jeremiah	Total: 6 points			NOT STARTED
	MEDIUM	Retrieve (user)	*Retrieve selected user data*		Kristyna	2 point			NOT STARTED
	MEDIUM	Retrieve (stock)	*Retrieve current stock data*		Nick	3 points			NOT STARTED
	MEDIUM	Compare	*Compare user expenditures and return applicable stock data*		John	1 point			NOT STARTED
	LOW	**Develop search for stocks**	Develop method to search for stock information by ticker	ST-16	Jeremiah	Total: 7 points			NOT STARTED
	LOW	Search stocks	*Allow query to search appropriate sections.*		Kristyna	5 points			NOT STARTED

LOW	Get information	*Use appropriate API to pull stock information.*		Nick	2 points			NOT STARTED	

7.2 Trello Project Management

Below is a screenshot for our project management system. Here you see some corresponding screenshots of that align with preliminary designs. Ways we are implementing this system is by using the features such as, assign members, create due date, add attachments, create sub tasks and activities to comment in a centralized way pertaining to only this large task. In the darkened background behind the document you will see the other sections that you are able to move each task to. For example, if there is a bug in the 'Design Login Page' you can move this section to that to notify everyone in the team. This is additionally a great way to have the client oversee the work progress.

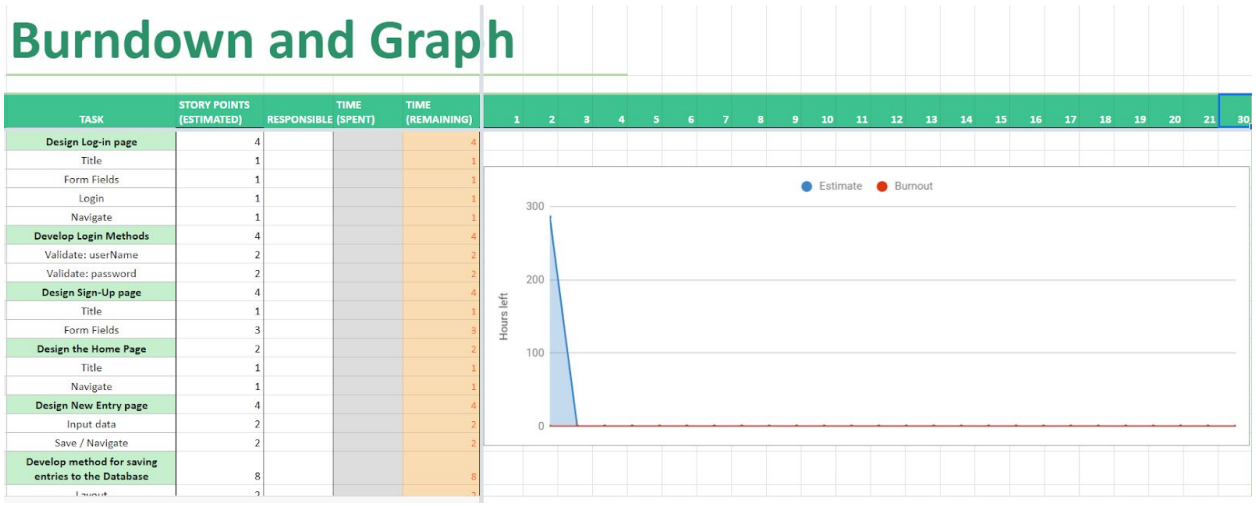
To view up-to-date information pertaining to this, please click here.



7.3 Burndown and Graph

The chart allows for the efficiency of analysing and keeping the workflow steady. The chart uses a SCRUM Agile method to show weekly work tasks. These tasks involve weekly tasks called sprints. We are able to use this graph to keep a daily chart of what tasks we are working on and how much time, or ‘story points’ we will complete by taking each task. The graph below is a representation of the estimate versus the burnout. The burnout is the amount of time the developers have taken to do this sprint. As you can see the burnout is flat because no tasks have been started.

To view a more clear version of the chart, the graph, and daily progress, please click here.



References

Anon, Alpha Vantage API Documentation. *API Documentation | Alpha Vantage*, https://www.alphavantage.co/documentation/

“9 Charts Showing Why You Should Invest Today.” *U.S. News & World Report*, U.S. News & World Report, https://money.usnews.com/investing/investing-101/articles/2018-07-23/9-charts-showing-why-you-should-invest-today.

Backman, Maurie. “You Don't Need That: Average American Spends Almost $18,000 a Year on Nonessentials.” *USA Today*, Gannett Satellite Information Network, 7 May 2019, https://www.usatoday.com/story/money/2019/05/07/americans-spend-thousands-on-nonessentials/39450207/

Backman, Maurie. “How Much Does Average US Household Have in a Savings Account?” *USA Today*, Gannett Satellite Information Network, 28 Sept. 2018, https://www.usatoday.com/story/money/personalfinance/budget-and-spending/2018/09/26/how-much-average-household-has-savings/37917401/.

Emmie Martin. “The Top 10 Things Young People Waste Money On.” *CNBC*, CNBC, 4 June 2019, https://www.cnbc.com/2017/06/06/72-percent-of-millennials-waste-money-on-dining-out.html.

Marsic, Ivan. *Book: Software Engineering - Textbook by Ivan Marsic*, https://www.ece.rutgers.edu/~marsic/books/SE/.