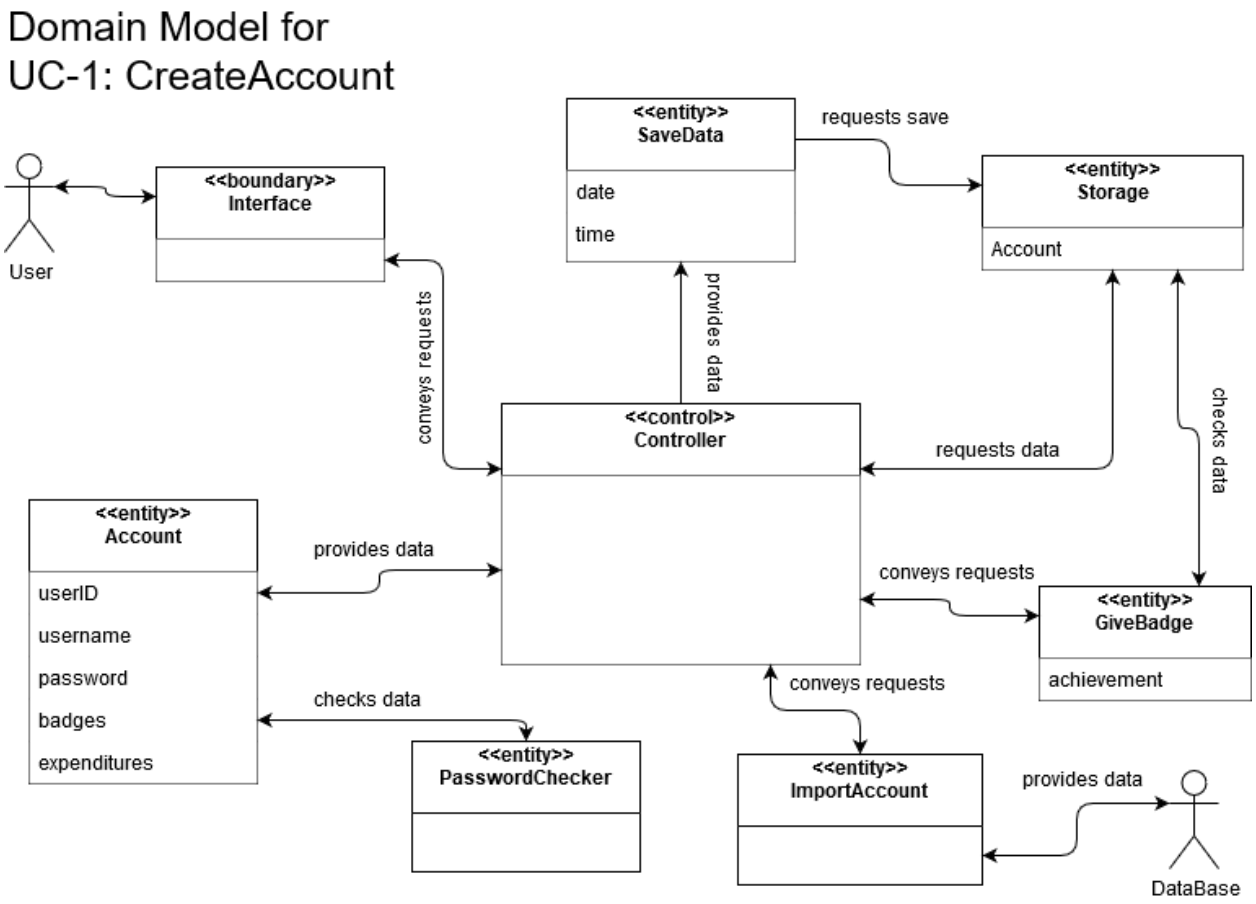
Domain Analysis

5.1 Domain Model

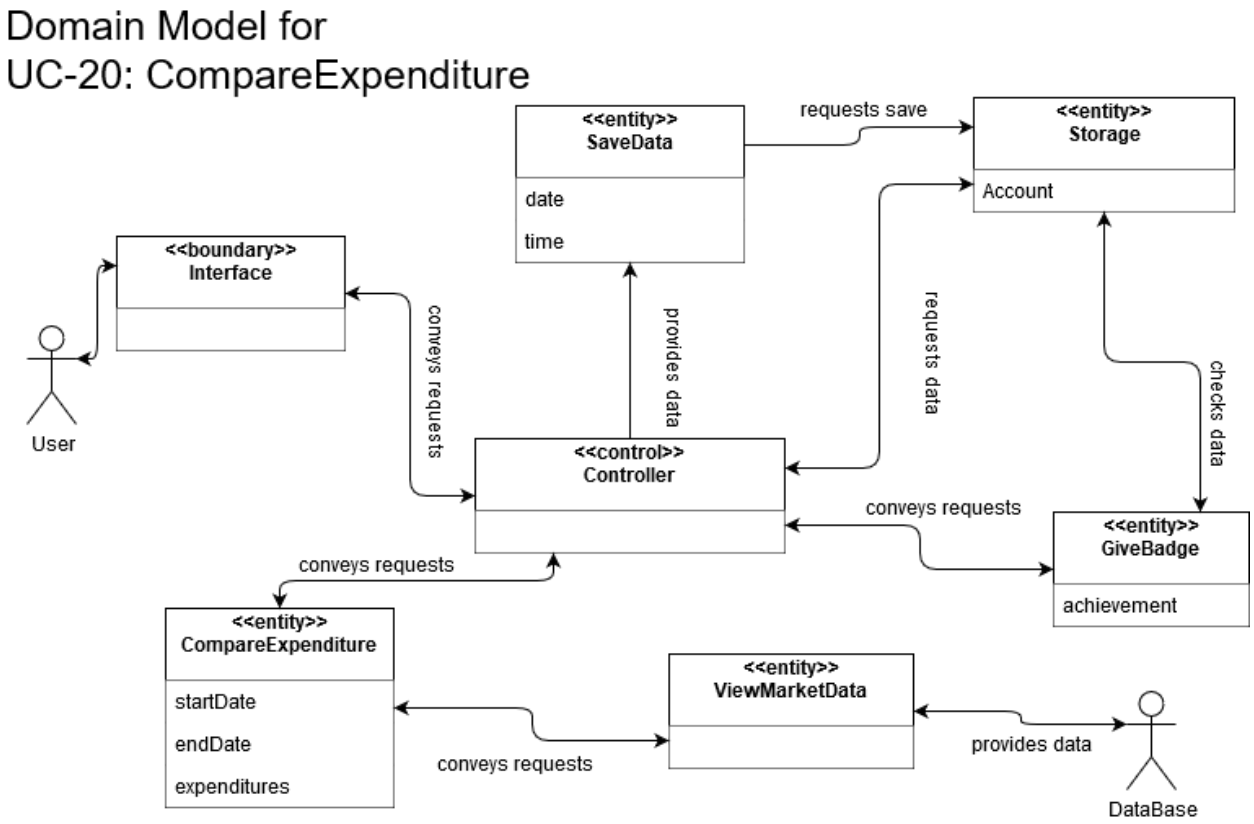
5.1.1 Concept Definitions

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.



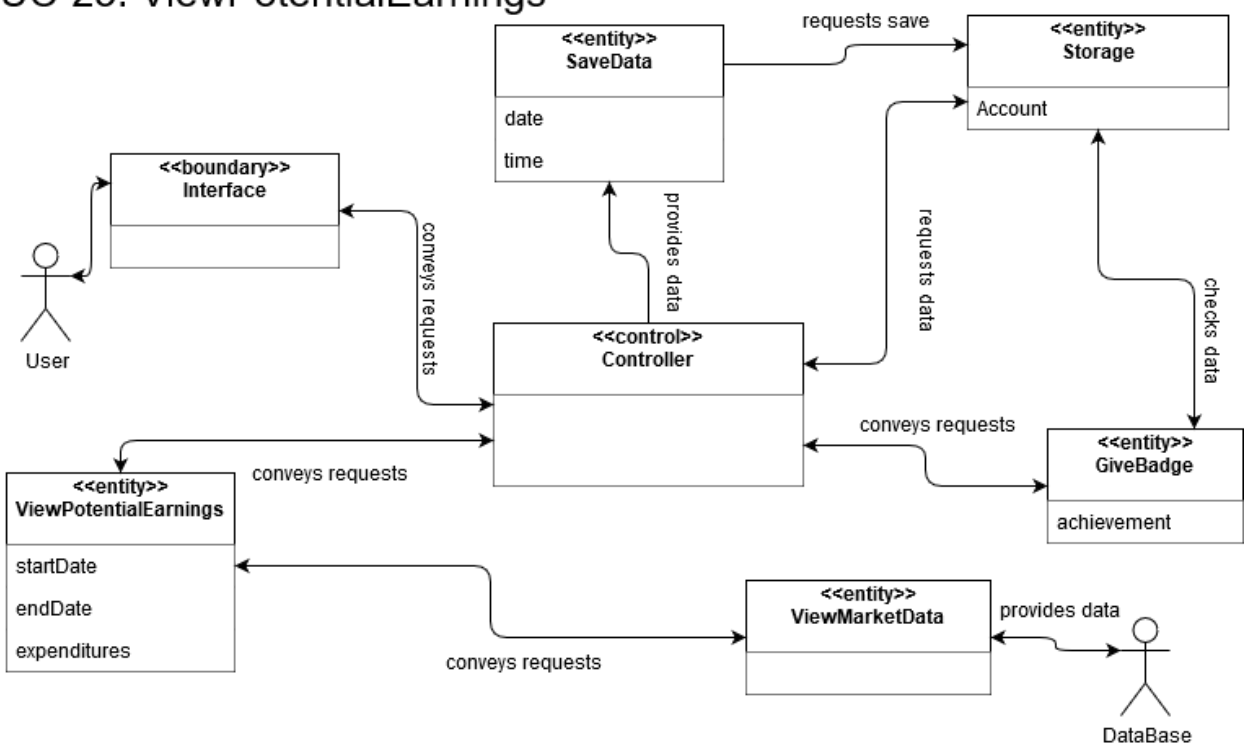
The diagram for UC-1 represents the user creating a new account with the InvestMe application. Initially, the user will interact with an interface and select the option to create a new account. When the user selects CreateAccount, the request is conveyed to the controller, who then delegates requests 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

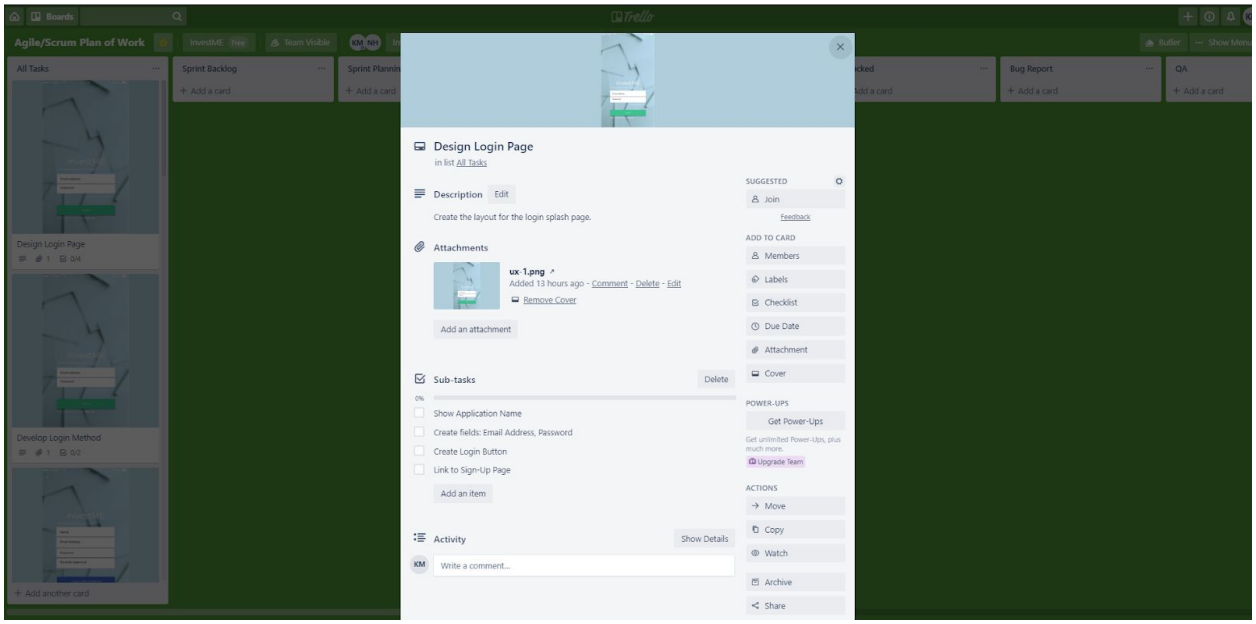
	LOW	Layout	*Create appropriate scaffolds for tutorial*		Kristyna	2 points			NOT STARTED
	LOW	Design Slides	*Create tutorial slideshow*		Nick	2 points			NOT STARTED
	LOW	Implement slides	*Write tutorial slide information*		John	2 points			NOT STARTED
	MEDIUM	**Develop method to compare stock data**	Compare 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.

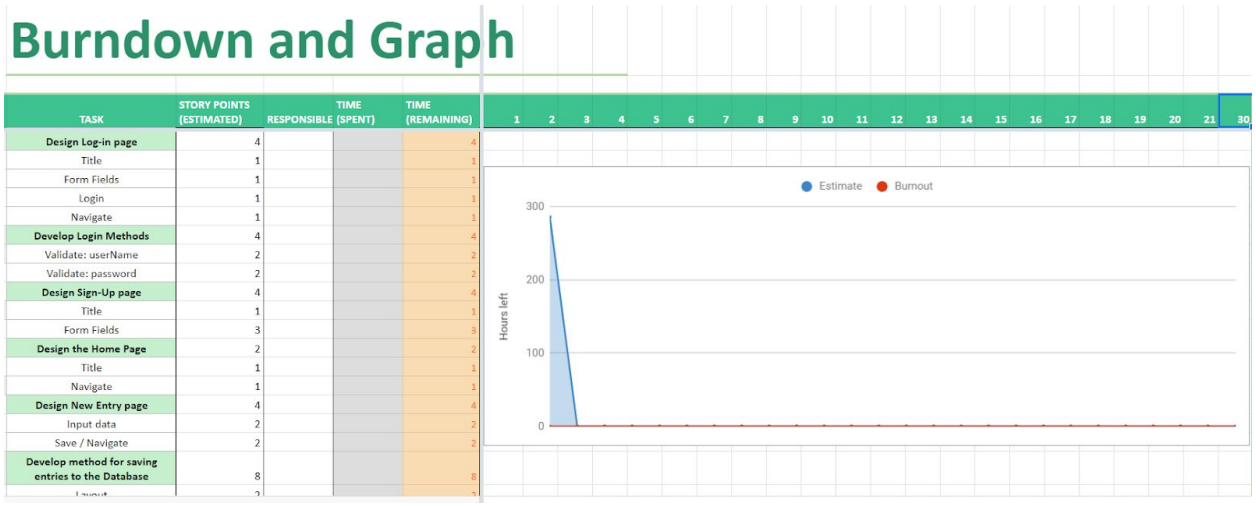
To view up-to-date information pertaining to this, please click here.



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/.