CSC4350/6350

Spring 2017

A stock trading software. The price is real-time and the exchange is virtual.

Virtual Stock

BATMAN

Mengyuan Zhu, Sungjae Kim, Sharon Kim, Jakub Pietrasik, Hyeun Kang

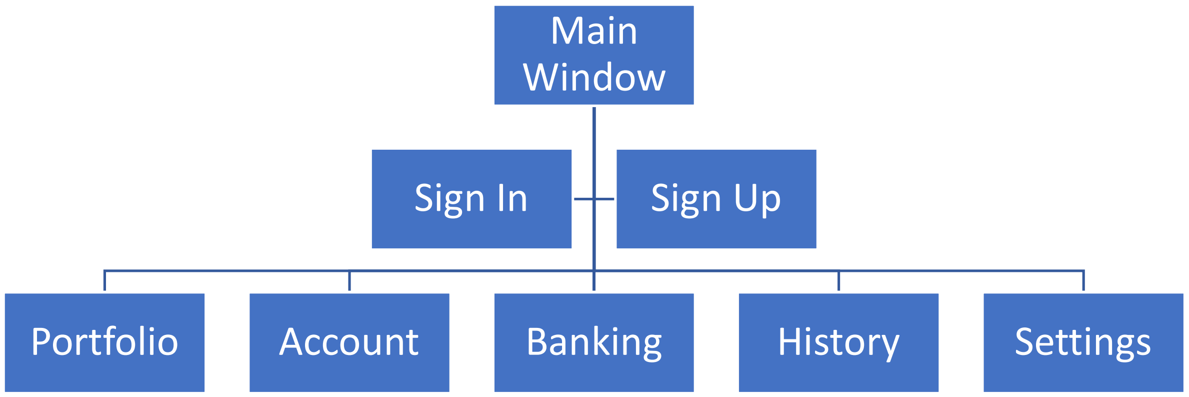
2/6/2017

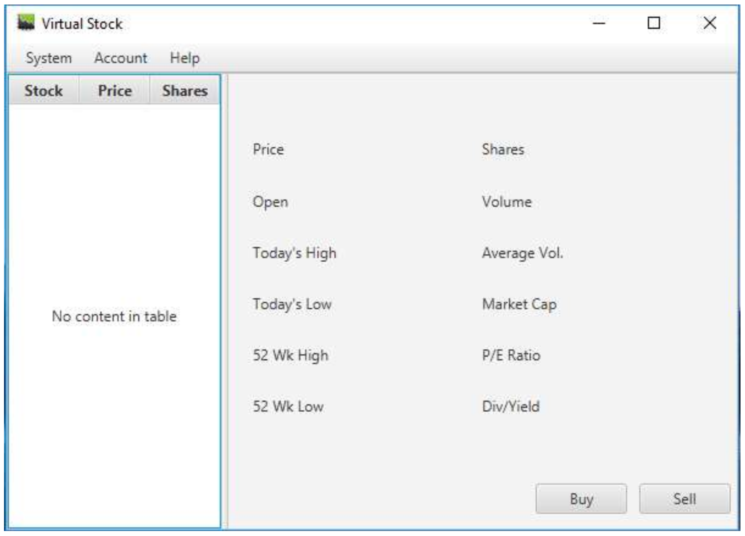
Virtual Stock

Contents

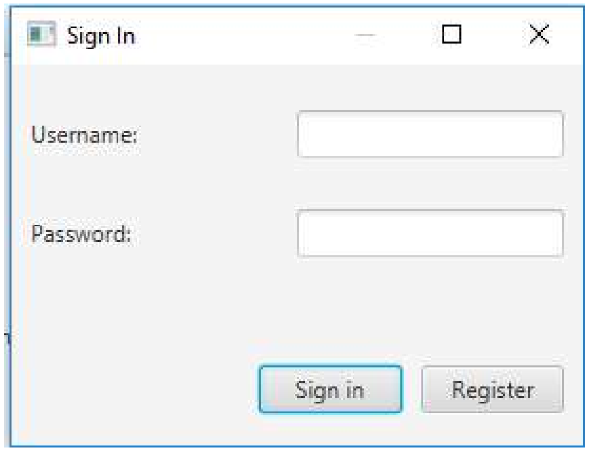
Horizontal Prototype

The following pages comprise Virtual Stock's horizontal prototype. The layout may be graphically represented as:

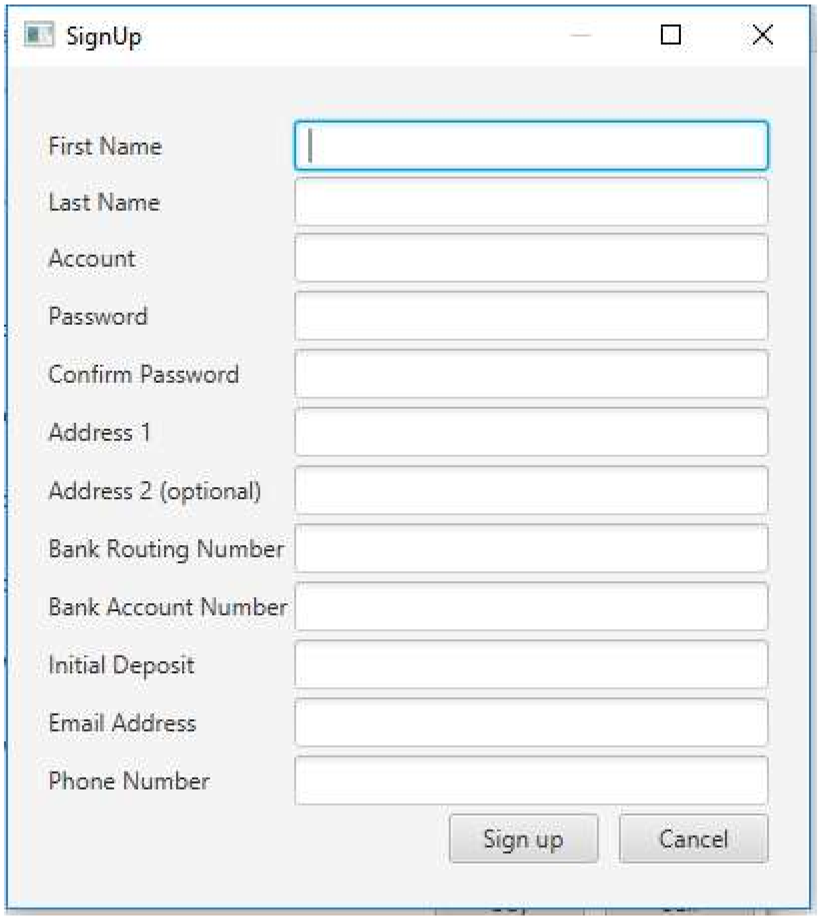




Main Window



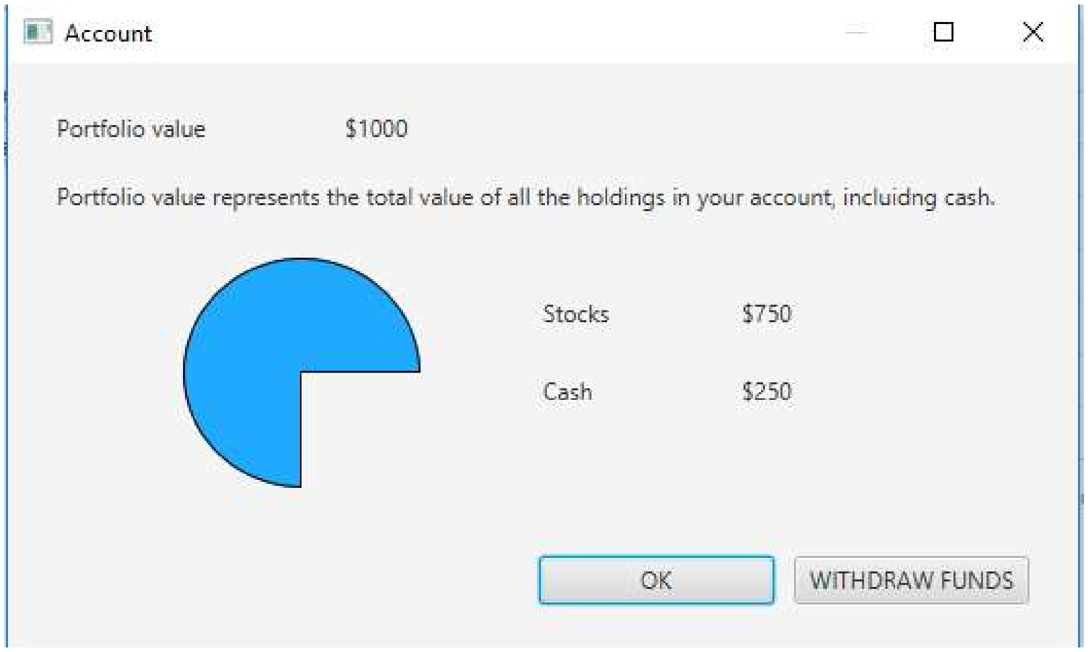
Sign In



SignUp

First Name	
Last Name	
Account	
Password	
Confirm Password	
Address 1	
Address 2 (optional)	
Bank Routing Number	
Bank Account Number	
Initial Deposit	
Email Address	
Phone Number	

Sign up



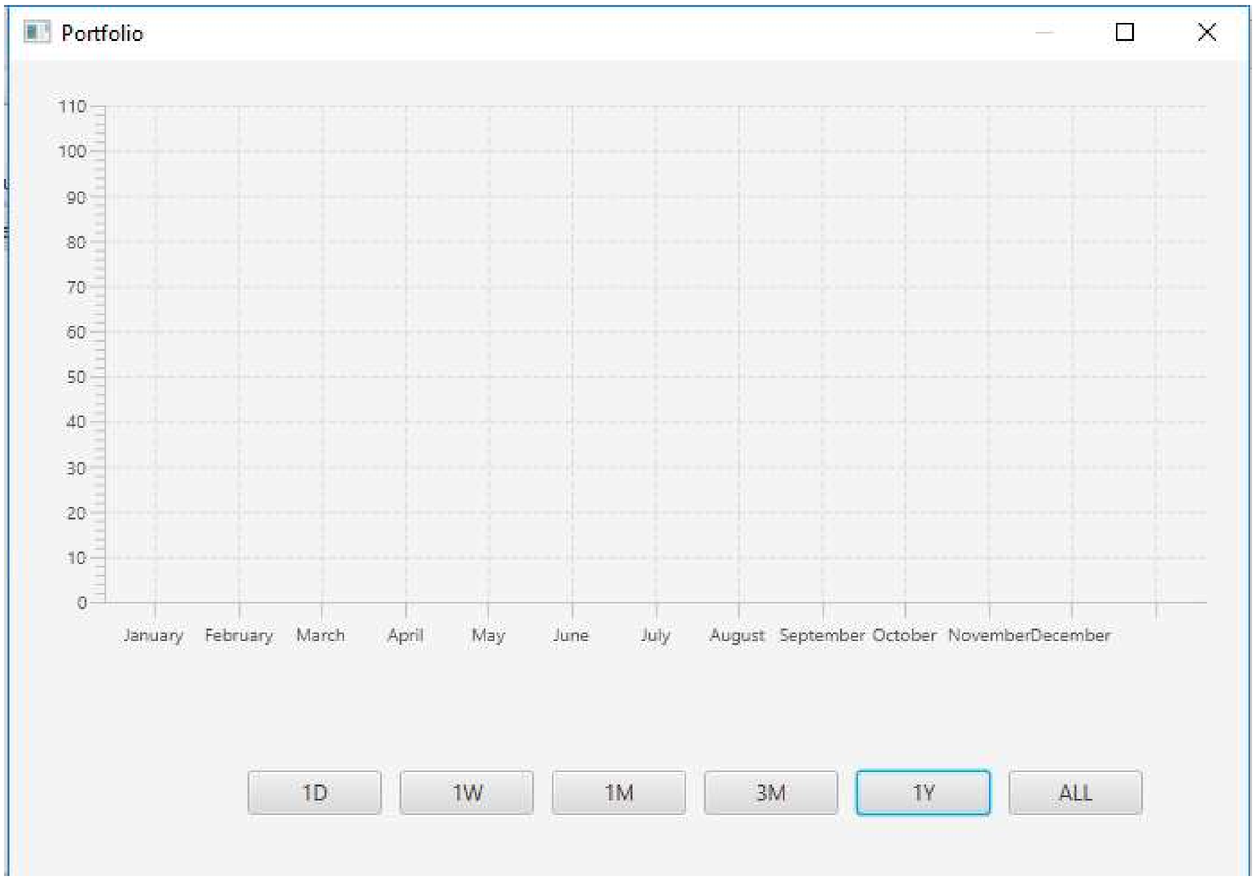
Account

Portfolio value $1000

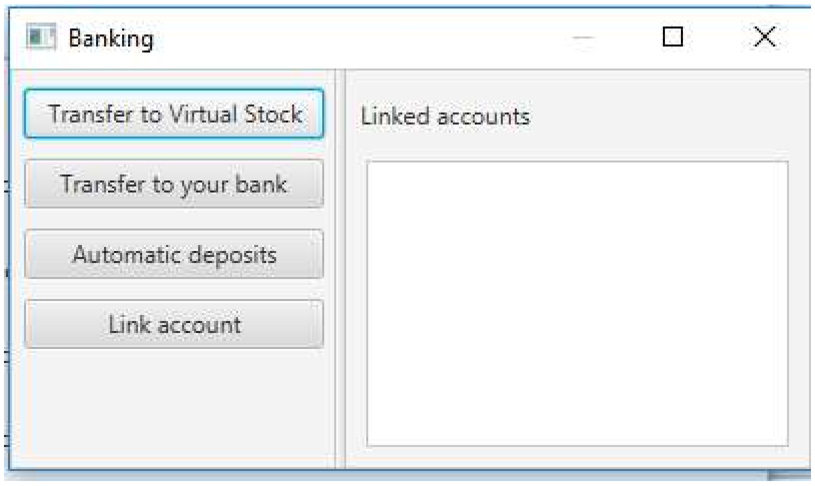
Portfolio value represents the total value of all the holdings in your account, incluidng cash.

Stocks	$750
Cash	$250

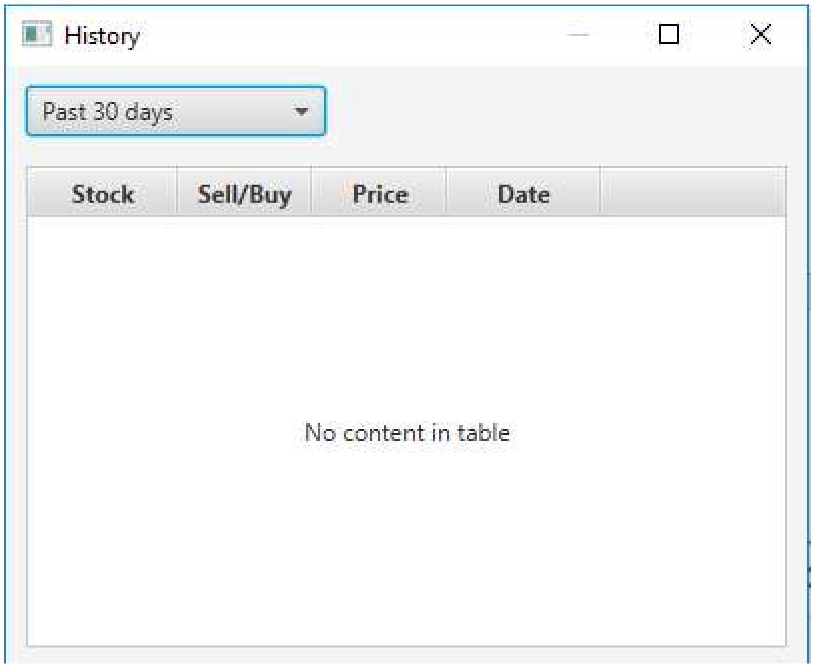
Account



Portfolio



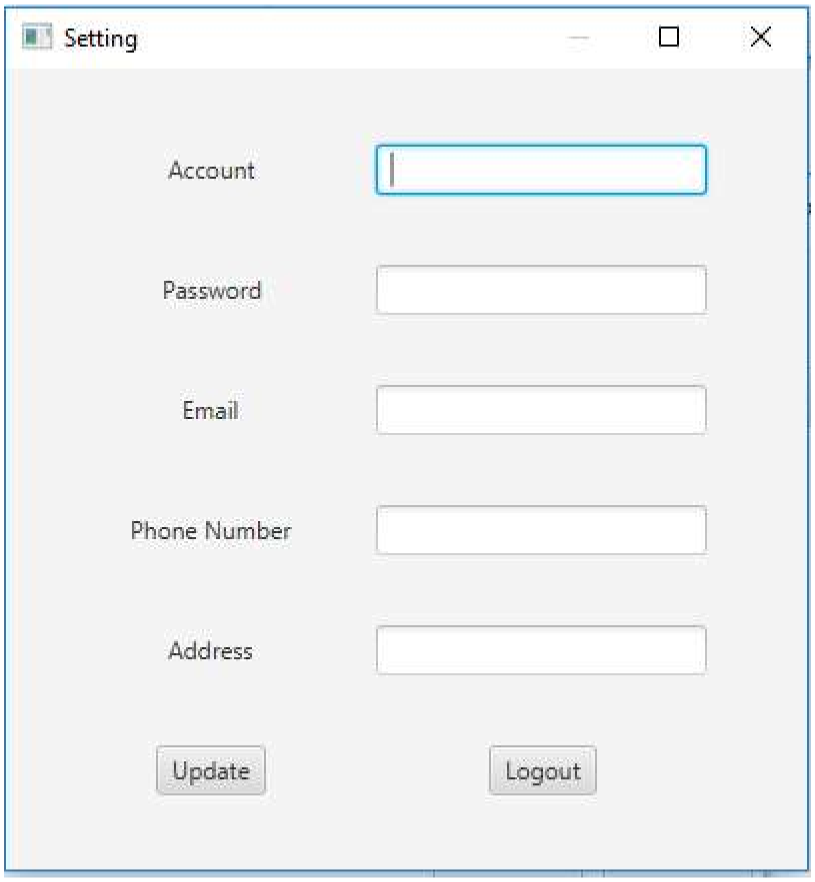
Banking



Stock	Sell/Buy	Price	Date

No content in table

History



Account	
Password	
Email	
Phone Number	
Address	

Settings

Problem Statements

Virtual Stock-1.0 Introduction

Batman shall develop a stock trading software with a user-friendly GUI. The software is PC based. The price is real-time from the Internet while the exchange is virtual. It shall be designed for beginners who want to learn stock trading.

Virtual Stock-2.0 User Registration

The system shall provide a window for user to register an account in Batman. Each user shall have his/her own account after registration. Users shall provide the following information: First Name, Last name, Address, Birthday, SSN, Bank account routing number and account number, Email address, Nationality and Agreement Form.

Virtual Stock-2.1

If a same account has been built up, the system shall let user know and ask the user to refill the form for registration

Virtual Stock-3.0 User Log In

The system shall have a window for user to log in.

Virtual Stock-3.1

The window shall provide a user name input, a password input, a forget password button, a sign-in button and a sign-up button.

Virtual Stock-3.2

The system shall let user into the "Portfolio" interface if the user type the right user name and password.

Virtual Stock-3.3

If the user failed to log in to the system, the system shall ask the user to refill the password.

Virtual Stock-3.4

The system shall provide a pin number for quick log in.

Virtual Stock-4.0 Portfolio

The system shall have a portfolio interface.

Virtual Stock-4.1

In this interface, the system shall show a chart of user's balance chart of today, a total balance value, a breaking news of today, stocks that a user keeps and his/her Shares, and a Watch list.

Virtual Stock-4.2

A search button shall also be shown in the corner for user to search a specific stock.

Virtual Stock-4.3

The color of the GUI shall be green if the user's balance goes up and be red if the user's balance goes down.

Virtual Stock-5.0 Account

The system shall provide a window to show user's account.

Virtual Stock-5.1

The account shall show the Total balance, Stocks balance and Cash balance.

Virtual Stock-6.0 Banking

The system shall give a function for user to transfer money to a bank or to the Batman app.

Virtual Stock-6.1

The linked accounts shall be shown in the bottom.

Virtual Stock-6.2

The system shall provide a bank account for user to deposit money.

Virtual Stock-7.0 History

The system shall provide a list of history of user's trading log. The list shows the deposit form a bank, the buying and selling with the corresponding money.

Virtual Stock-7.1

The system shall also show the date.

Virtual Stock-7.2

The system shall allow the user to see the history from/to a specific date.

Virtual Stock-8.0 Setting

The system shall have a setting interface for user to reset pin number and update user information including name, password, email, phone number and address.

Virtual Stock-8.1

It shall also have a log out button for user to log out.

Requirements Traceability Matrix

Entry#	Para#	Requirements Traceability Matrix	Type	Use Case Name
1	1	Batman shall develop a stock trading software with a user-friendly GUI.	SW, HW	Use Case 3 Show_Main_Window

2	2	The system shall provide a window for user to register an account in Batman.	SW	Use Case 2 User_Sign_Up
3	2.1	The system shall let user know and ask the user to refill the form for registration.	SW	n/a
4	3	The system shall have a window for user to log in.	SW	Use Case 1 User_Sign_In
5	3.1	The window shall provide a user name, password input, forgot password button, and sign in and sign up buttons.	SW	n/a
6	3.2	The system shall let user into the interface if the user type the right user information	SW	n/a
7	3.3	If the user failed to log in to the system, the system shall ask the user to refill the password.	SW	n/a
8	3.4	The system shall provide a pin number for quick log in	NTH	n/a
9	4	The system shall have a portfolio interface.	SW	Use Case 4 Show_Portfolio
10	4.1	The system shall show a chart of user's balance chart of today, total balance value, breaking news of today, stocks that a user keeps and his/her shares, and a watch list.	NTH	n/a
11	4.2	The search button shall also be shown in the corner for user to search a specific stock.	NTH	n/a
12	4.3	The color of the GUI shall be green if the user's balance goes up and be red if the user's balance goes down.	NTH	n/a
13	5	The system shall provide a window to show user's account.	SW	Use Case 5 Show_Account
14	5.1	The account shall show the total balance, stocks balance, and cash balance.	NTH	n/a
15	6	The system shall give a function for user to transfer money to a bank or to the Batman app.	SW	Use Case 6 Show_Banking
6	6.1	The linked accounts shall be shown in the bottom.	NTH	n/a
17	6.2	The system shall provide a bank account for user to deposit money.	SW	n/a
18	7	The system shall provide a list of history of user's trading log.	SW	Use Case 7 Show_History
19	7.1	The system shall also show the date.	NTH	n/a
20	7.2	The system shall allow the user to see the history from/to a specific date.	NTH	n/a

21	8	The system shall have a setting interface for user to reset pin number and update user information including name, password, email, phone number and address.	SW	Use Case 8 Show_Settings
22	8.1	The system shall also have a log out button for the user to log out.	SW	n/a

Use cases (rational) and Interaction Diagrams

Use Case 1: User_Sign_In

Overview:
This Use Case enables existing Users to log into their own account and allows new Users to create a Virtual Stock account.

Rationale:
User accounts personalize Virtual Stock usage so that a User can reuse many Virtual Stock features customized to the User's needs. These User accounts are also necessary to implement the Virtual Stock protocol function. With those aspects in mind, this Use Case ensures that a User has a Virtual Stock account before using Virtual Stock's main functions.

Preconditions:
1. User has opened the Virtual Stock application successfully.
2. User has not signed in.

Scenario:

Action	Software Reaction
1. User runs the Virtual Stock application.	1. Sign In View will display the Sign In Form and prompt the User to enter Username and Password, or create a new account.
2. User fills out the Username and Password fields and submits form.	2. Sign In View checks form info in the User Database and finds match; displays MainWindow View.
3. User fills out the Username and Password fields and submits form.	3. Sign In View checks form info in the User Database and finds no match; clears form and prompts User to resubmit.
4. User clicks the Sign up button	4. User_Sign_Up invoked.

Scenario Notes:
2 and 3 are mutually exclusive with 4. In other words, a User cannot sign into an existing account and create a new account during User_Signs_In. 3-5 occur in order, but can be broken if the User signs in correctly with Username and Password.

Post Conditions:
1. Home View is displayed.
2. User Profile information is loaded.

Required GUI:
SignIn

Use Cases Utilized:
User_Creates_Account

Timing Constraints:
None

Use Case 2: User_Sign_Up

Overview:
This Use Case enables the User to create new account.

Rationale:
This Use Case's rationale directly follows from the rationale given in UC1.

Preconditions:
1. User has opened the Virtual Stock application successfully.
2. User has not signed in.

Scenario:

Action	Software Reaction
1. User runs the Virtual Stock application.	1. Sign In View will display the Sign In Form and prompt the User to enter Username and Password, or create a new account.
2. User clicks Sign Up	2. Sign In View prompts User to enter a unique Username and password.
3. User enters all text fields entry.	3. Sign In View will check basic format, and also check whether this Username exists. If it exits, message that Username already existed, clear the Username field.
4. User presses Sign up Button	4. If both Username and Password fields are filled and correct, Sign In View adds account and displays Home View with default Profile. If incorrect, message that at least one field is not filled in correctly is displayed.
5. User presses Cancel button.	5. Sign In View displays Sign In Form.

Scenario Notes:
Items 1-4 must be done in order. Items 5 are mutually exclusive.

Post Conditions:
The User has a Virtual Stock account and Home View is displayed with default profile.

Required GUI:
MainWindow, SignIn

Exceptions:
1. Username entered already exists.
2. Username does not meet format requirements.
3. Password does not meet format requirements.

Use Cases Utilized:
None

Timing Constraints:
None.

Use Case 3: Show_Main_Window

Overview:
This Use Case enables Users to see the main window

Rationale:
By showing main window, users have access to see several other windows. Such as stock name, buying/selling stocks, access to portfolios, personal account, banking account, history, settings, about, and exiting the program.

Preconditions:
1. User has signed in.
2. MainWindow is displayed.
3. User has connected to the Internet.

Scenario:

Action	Software Reaction
1. User clicks stock name	1. Virtual stock will show the details of the stocks including price, shares, open, volume, today's high, average volume today's low, market capacity, 52 week high, 52 week low, P/E ratio and Div/Yield.
2. User clicks buy	2. A window will show the price and there will and input for user to type the number of shares to be purchased. User's remaining cash can also be shown out.
3. User clicks sell	3. A window will show and asks the user how many shared a user wants to sell.
4. User clicks Account - Portfolio	4. Portfolio window shows up
5. User clicks Account - Account	5. Account window shows up
6. User clicks Account - Banking	6. Banking window shows up
7. User clicks Account - History	7. History window shows up
8. User clicks Account - Settings	8. Settings window shows up
9. User clicks Help - About	9. About window shows up
10. User clicks File - Exit	10. Program exits

Scenario Notes:
None

Post Conditions:
1. If User has made updates and saved, the User's stock is saved successfully, and MainWindow displays the current shares of users' stocks
2. If User cancels, MainWindow displays the original stocks the user has.

Required GUI:
MainWindow, StockPurchase, StockSell, Portfolio, Account, Banking, History, Settings, About

Exceptions:	**Use Cases Utilized:**	**Timing Constraints:**
None	None	None

Use Case 4: Show_Portfolio

Overview:
This Use Case is to enable Users to see the portfolio window.

Rationale:
By showing portfolio, users have access to see days, weeks, months, and years worth of stocks.

Preconditions:
1. User is logged into Virtual Stock.
2. MainWindow is displayed.
3. User clicks Account-Portfolio.

Scenario:

Action	Software Reaction
1. User clicks 1D.	1. Chart gives 1 day data of user' stocks
2. User clicks 1W.	2. Chart gives 1 week data of user' stocks
3. User clicks 1M.	3. Chart gives 1 month data of user' stocks
4. User clicks 3M.	4. Chart gives 3 months data of user' stocks
5. User clicks 1Y.	5. Chart gives 1 year data of user' stocks
6. User clicks ALL.	6. Chart gives all data of user' stocks

Scenario Notes:
Items 1-6 are mutually exclusive.

Post Conditions:
The user will be returned to the MainWindow.

Required GUI:
Portfolio

Exceptions:
None

Use Cases Utilized:
Show_Main_Window

Timing Constraints:
None.

Use Case 5: Show_Account

Overview:
This Use Case it to enable Users to show user's account information including the total money, stock and cash percentage. User can also withdraw funds from this window.

Rationale:
By showing account, users have easy access to their money and funds which they can withdraw from their virtual stock to their bank.

Preconditions:
1. User is logged into Virtual Stock.

2. MainWindow is displayed.
3. User clicks Account-Account.

Scenario:

Action	Software Reaction
1. User clicks OK.	1. Window closes
2. User clicks Withdraw Funds.	2. A window will show up and ask the user how much funds the user wants to withdraw.

Scenario Notes:
It the user types more money than he/she has, the system will not withdraw the funds from virtual stock to his/her bank and asks the user to reinput all the numbers.

Post Conditions:
The user will be returned to the MainWindow.

Required GUI:
Account

Exceptions:
None

Use Cases Utilized:
Show_Main_Window

Timing Constraints:
None.

Use Case 6: Show_Banking

Overview:
This Use Case enables Users to transfer money to bank, transfer money from bank to virtual stock, automatic deposits and link account. The user can also find the linked bank account(s).

Rationale:
By showing bank account, the user has easy access to deposit money at a specific time to a bank or virtual stock by routing number and account number.

Preconditions:
1. User is logged into Virtual Stock.
2. MainWindow is displayed.
3. User clicks Account-Banking.

Scenario:

Action	Software Reaction
1. User clicks “Transfer to virtual stock”	1. The software will pop up a window and ask the user to select a bank account and money to be transferred.
2. User clicks “Transfer to bank”	2. The software will pop up a window and ask the user to select a bank account and money to be transferred.

3. User clicks “Automatic deposit”	3. The software will pop up a window and ask the user to select the time and money to be transferred from bank account to virtual stock
4. User clicks “Link account”	4. The software will pop up a window and ask the user to select a bank account and input the routing number and account number.

Scenario Notes:
1. There must be at least one bank account in user’s banking page.
2. There must be enough funds for user to transfer.

Post Conditions:
The user will be returned to the MainWindow.

Required GUI:
Banking

Exceptions:
None

Use Cases Utilized:
Show_Main_Window

Timing Constraints:
None.

Use Case 7: Show_History

Overview:
This Use Case enables Users to see the history records. It will also allow user to select the date from a specific time.

Rationale:
By showing history, users can see specific dates of stocks and their prices of when they are bought or sold

Preconditions:
1. User is logged into Virtual Stock.
2. MainWindow is displayed.
3. User clicks Account-History.

Scenario:

Action	Software Reaction
1. User selects a date range from the comboBox	1. The system will show the history record during this time. This includes stock name, sell or buy, price and date.

Scenario Notes:
None

Post Conditions:
The user will be returned to the MainWindow.

Required GUI:
History

Exceptions:
None

Use Cases Utilized:
Show_Main_Window

Timing Constraints:
None.

Use Case 8 Show_Settings

Overview:
This Use Case enables Users to see and edit settings

Rationale:
By showing settings, users can have the option to update or cancel editing personal info such as his/her account, email, password, phone number, and address.

Preconditions:
1. User is logged into Virtual Stock.
2. MainWindow is displayed.
3. User clicks Account-Portfolio.

Scenario:

Action	Software Reaction
1. User edits account	1. Account name changes
2. User edits email	2. Email address changes
3. User edits password	3. User password changes
4. User edits phone number	4. user phone number changes
5. User edits address	5. User address changes
6. User edits Update button	6. All changes saved
7. User edits Cancel button	7. All changes discarded

Scenario Notes:
None

Post Conditions:
The user will be returned to the MainWindow.

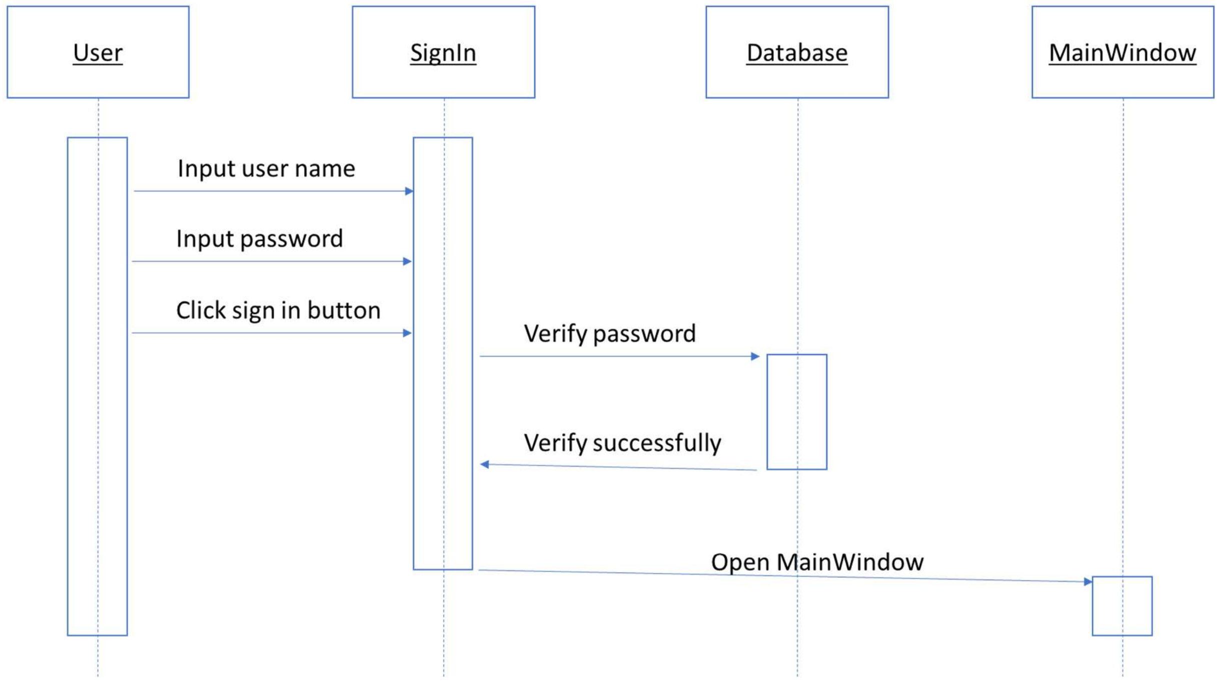
Required GUI:
Settings

Exceptions:
None

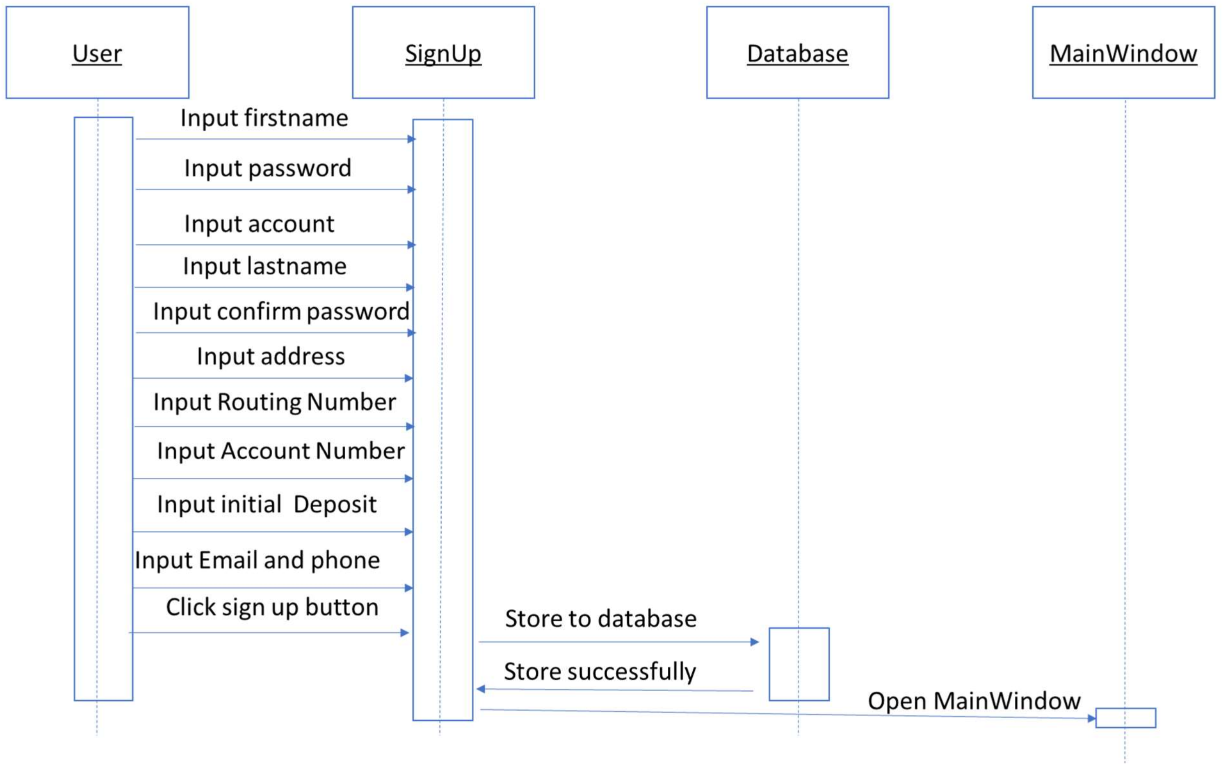
Use Cases Utilized:
Show_Main_Window

Timing Constraints:
None.

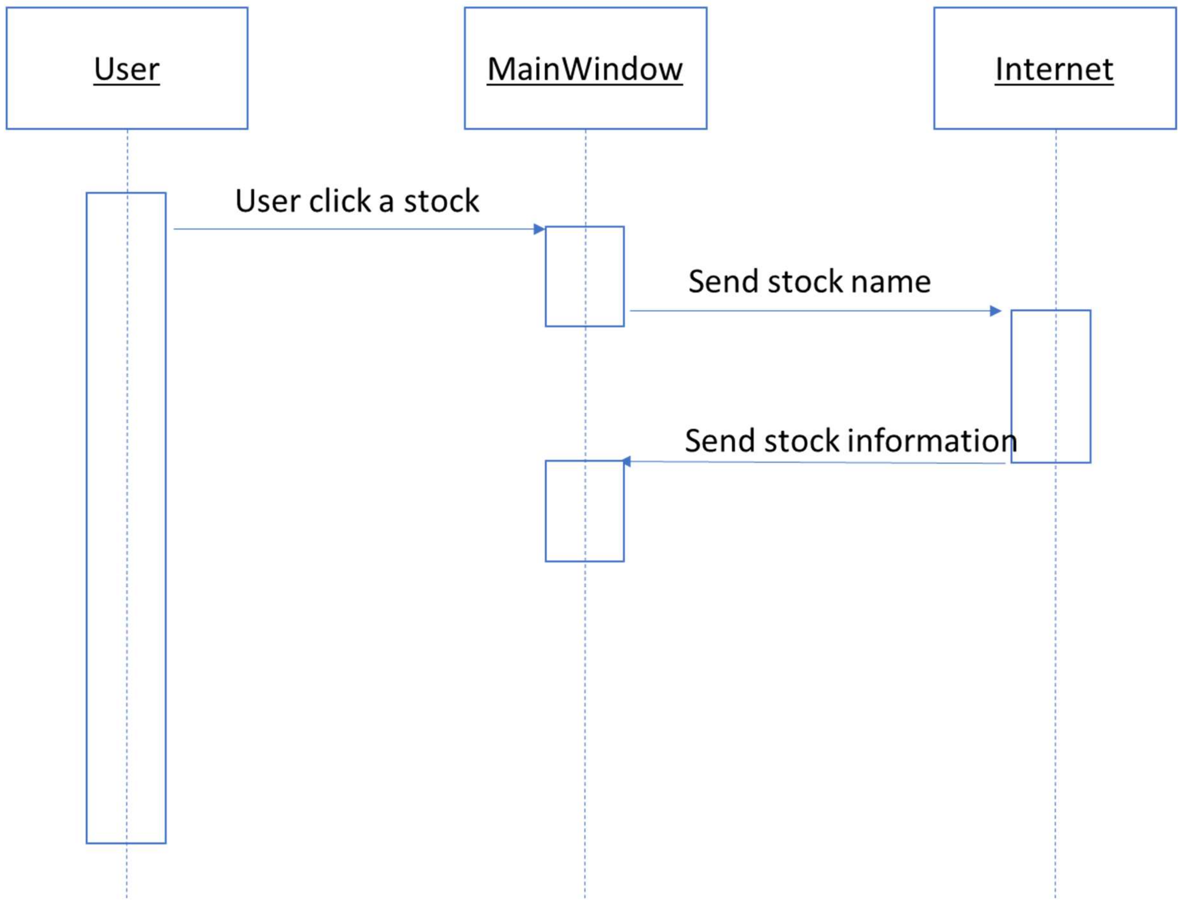
Use Case 1: User_Sign_In



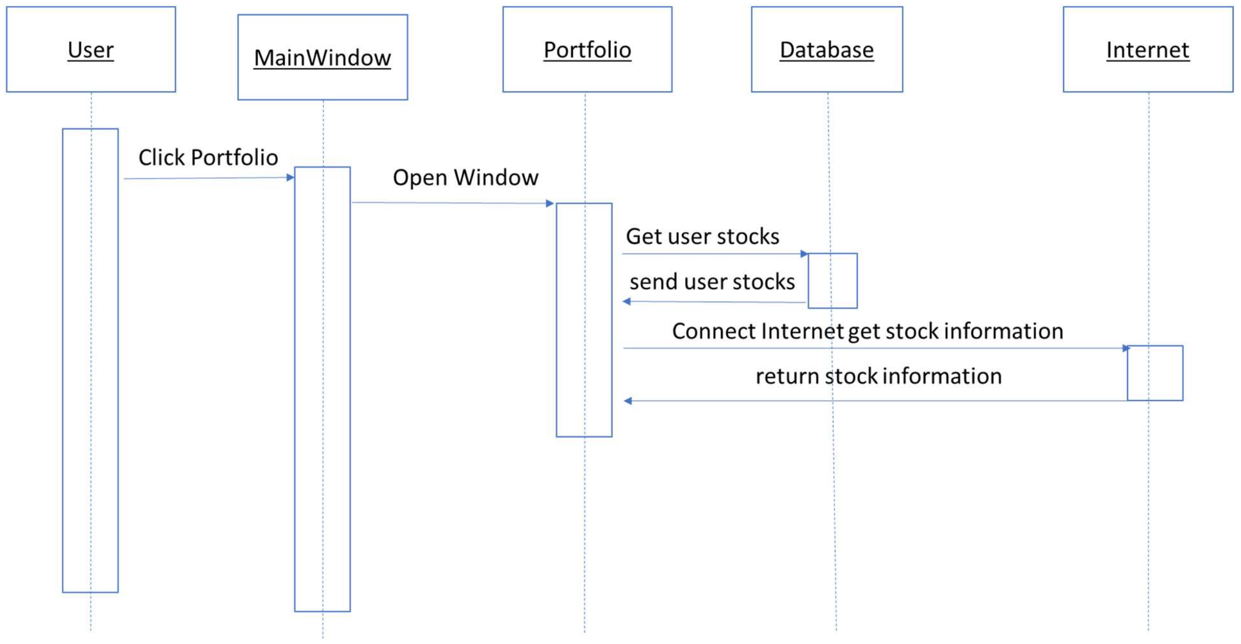
Use Case 2: User_Sign_Up



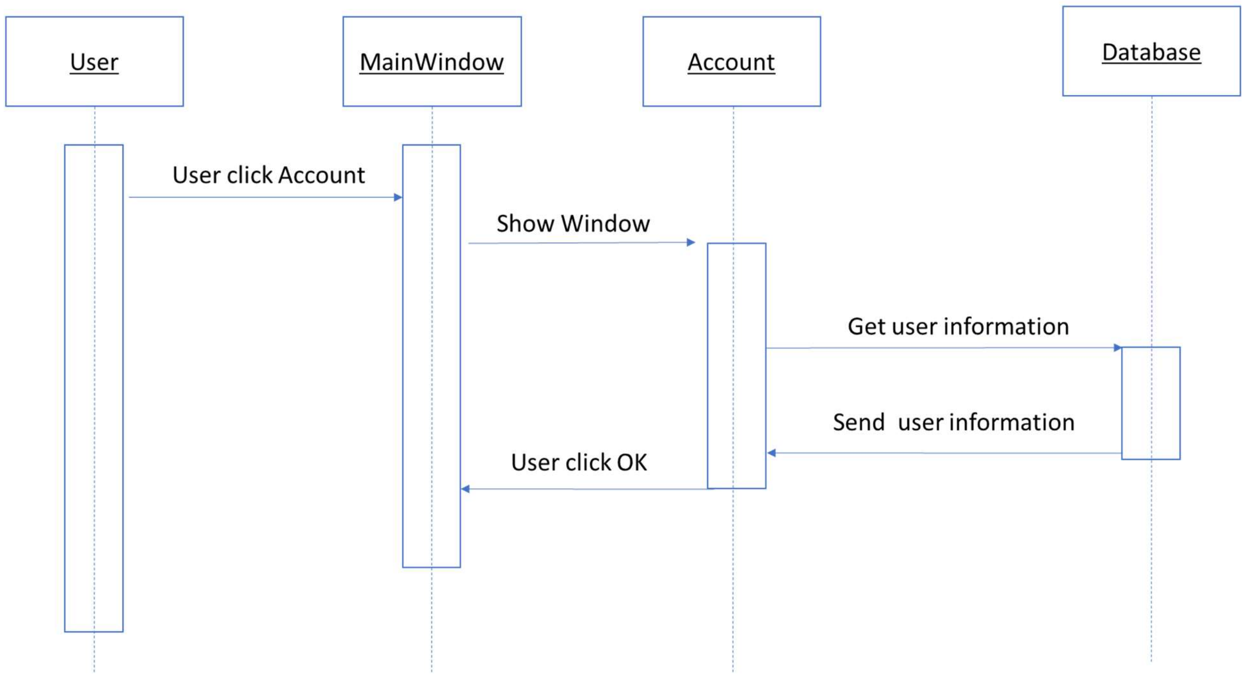
Use Case 3: Show_Main_Window



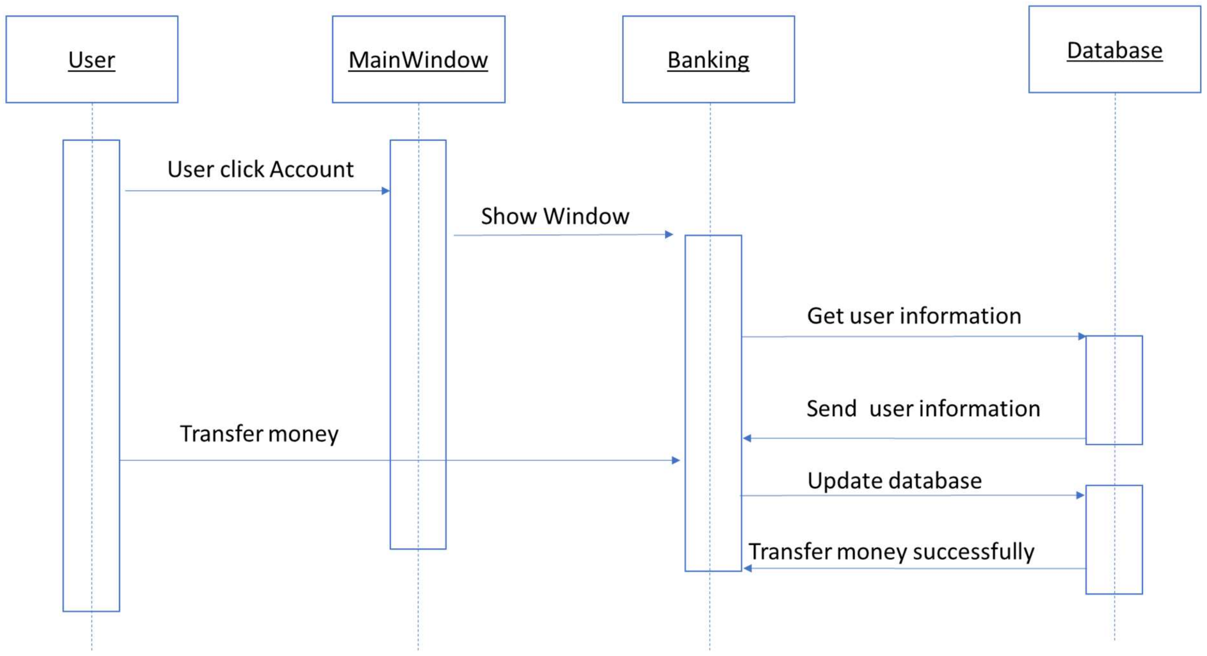
Use Case 4: Show_Portfolio



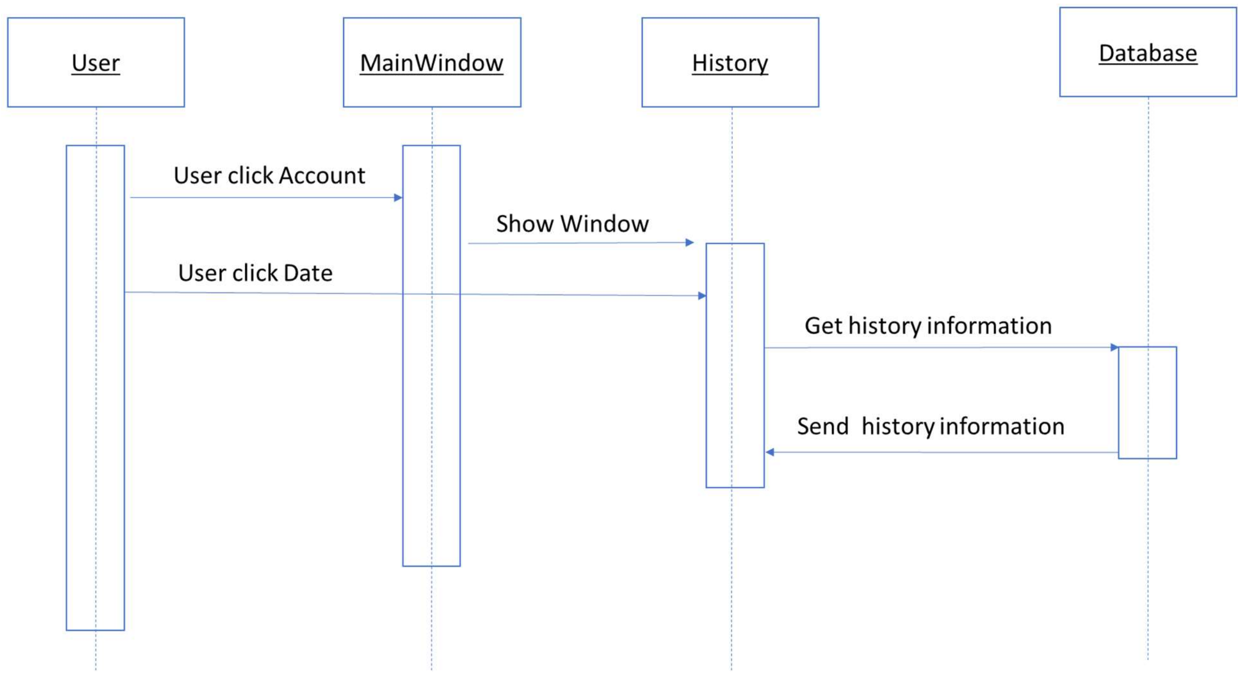
Use Case 5: Show_Account



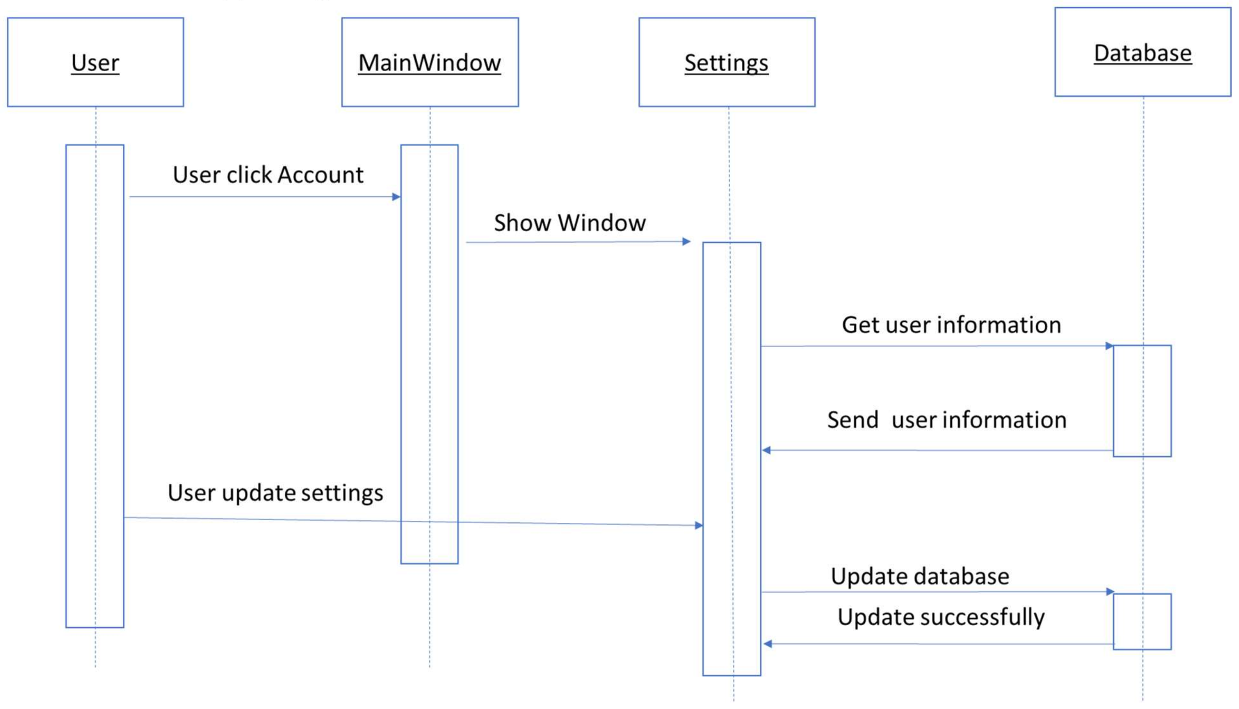
Use Case 6: Show_Banking



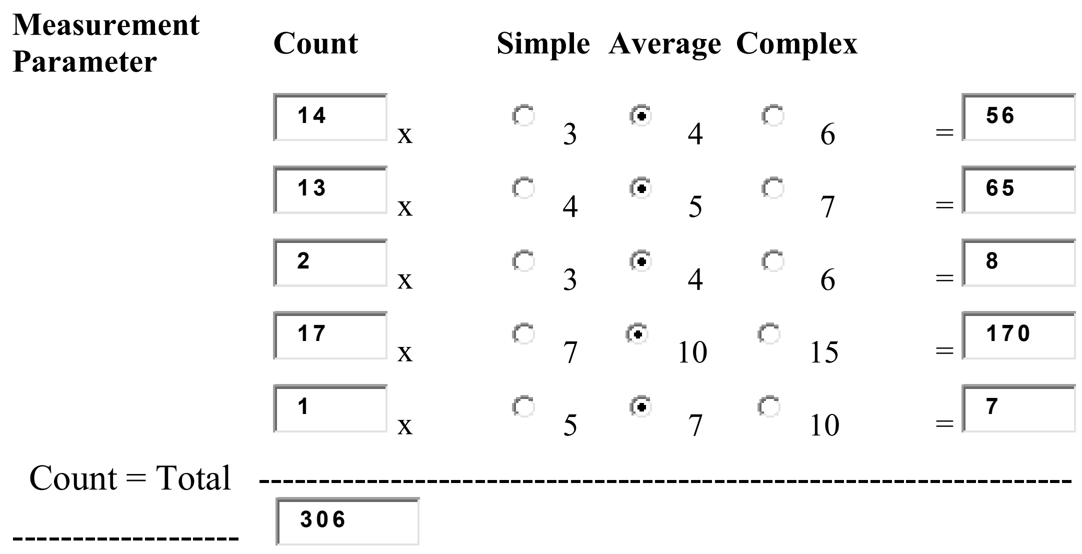
Use Case 7: Show_History



Use Case 8 Show_Settings



Function Point Cost Analysis



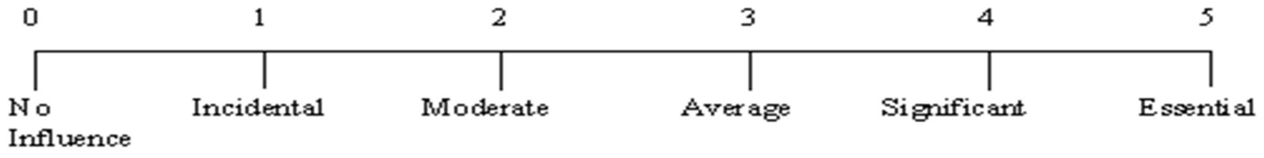
Measurement Parameter	Count		Simple	Average	Complex		
	14	x	○ 3	● 4	○ 6	=	56
	13	x	○ 4	● 5	○ 7	=	65
	2	x	○ 3	● 4	○ 6	=	8
	17	x	○ 7	● 10	○ 15	=	170
	1	x	○ 5	● 7	○ 10	=	7

Count = Total --

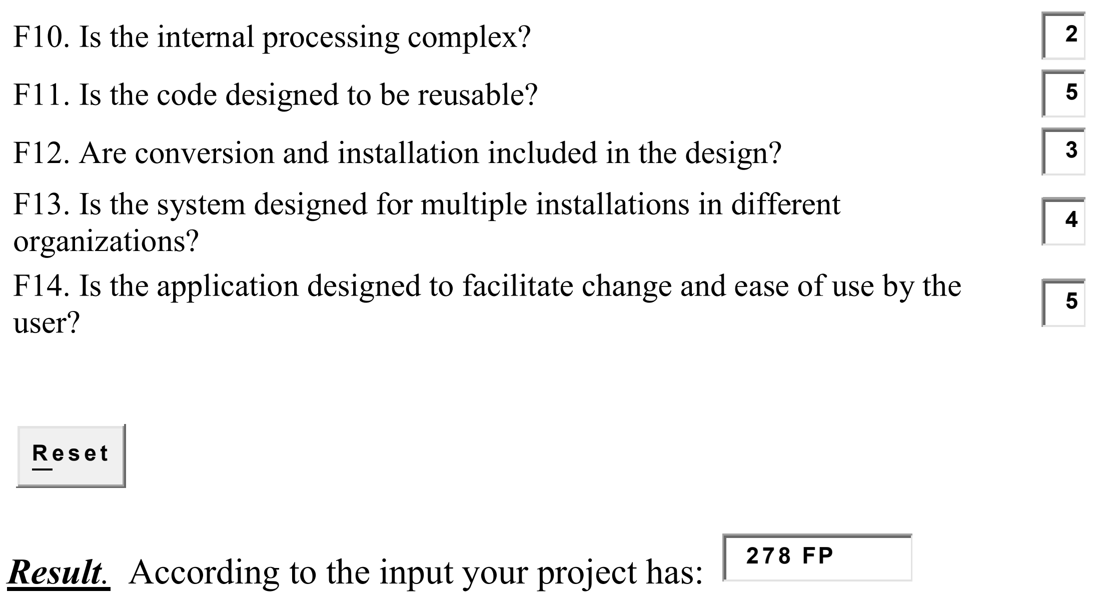
------------------ 306

Note. By clicking on the buttons above more information about the measurement parameters will be available.

Rate each factor (Fi, i=1 to14) on a scale of 0 to 5:



F1. Does the system require reliable backup and recovery?	0
F2. Are data communications required?	2
F3. Are there distributed processing functions?	2
F4. Is performance critical?	0
F5. Will the system run in a existing, heavily utilized operational environment?	0
F6. Does the system require on-line data entry?	0
F7. Does the on-line data entry require the input transaction to be built over multiple screens or operations?	1
F8. Are the master files updated on-line?	0
F9. Are the inputs, outputs, files or inquiries complex?	2



F10. Is the internal processing complex?	2
F11. Is the code designed to be reusable?	5
F12. Are conversion and installation included in the design?	3
F13. Is the system designed for multiple installations in different organizations?	4
F14. Is the application designed to facilitate change and ease of use by the user?	5

Result. According to the input your project has: 278 FP

Database to be used

XML

Work Structure Document

Group name: **Batman**

Name	**Tasks**
Mengyuan Zhu	Team Coordinator Documents handler Java coder Problem Statement Requirements Traceability Matrix Dictionary A horizontal prototype of the software to be developed Use cases and Interaction Diagrams - example as per given in class Database to be used
Sungjae Kim	Finalize code documentation Java coder Requirements Traceability Matrix Gannt Chart
Sharon Kim	User Guide Function Point Cost Analysis. Program tester Rationale
Jakub Pietrasik	Program tester
Hyeun Kang	Program tester

Dictionary

Portfolio: In finance, a portfolio is a collection of investments held by an investment company, hedge fund, financial institution or individual.

Broker: A person who buys or sells an investment for you in exchange for a fee (a commission). Here is Tim's favorite broker.

Dividend: this is a portion of a company's earnings that is paid to shareholders, or people that own hat company's stock, on a quarterly or annual basis. Not all company's do this.

Exchange: An exchange is a place in which different investments are traded. The most well-known in the United States are the New York Stock Exchange and the Nasdaq.

Quote: Information on a stock's latest trading price. This is sometimes delayed by 20 minutes unless you are using an actual broker trading platform.

Volume: The number of shares of stock traded during a particular time period, normally measured in average daily trading volume.

Yield: This usually refers to the measure of the return on an investment that is received from the payment of a dividend. This is determined by dividing the annual dividend amount by the price paid for the stock. If you bought stock XYZ for $40-a-share and it pays a $1.00-per-year dividend, you have a "yield" of 2.5%.

JDK: The Java Development Kit (JDK) is an implementation of either one of the Java Platform, Standard Edition; Java Platform, Enterprise Edition or Java Platform, Micro Edition platforms released by Oracle Corporation in the form of a binary product aimed at Java developers on Solaris, Linux, Mac OS X or Windows.

GUI: The graphical user interface is a type of user interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, instead of text-based user interfaces, typed command labels or text navigation.

HTTP: The Hypertext Transfer Protocol (HTTP) is an application protocol for distributed, collaborative, and hypermedia information systems. HTTP is the foundation of data communication for the World Wide Web.

XML: In computing, Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The W3C's XML 1.0 Specification and several other related specifications-all of them free open standards—define XML.

Gantt Chart

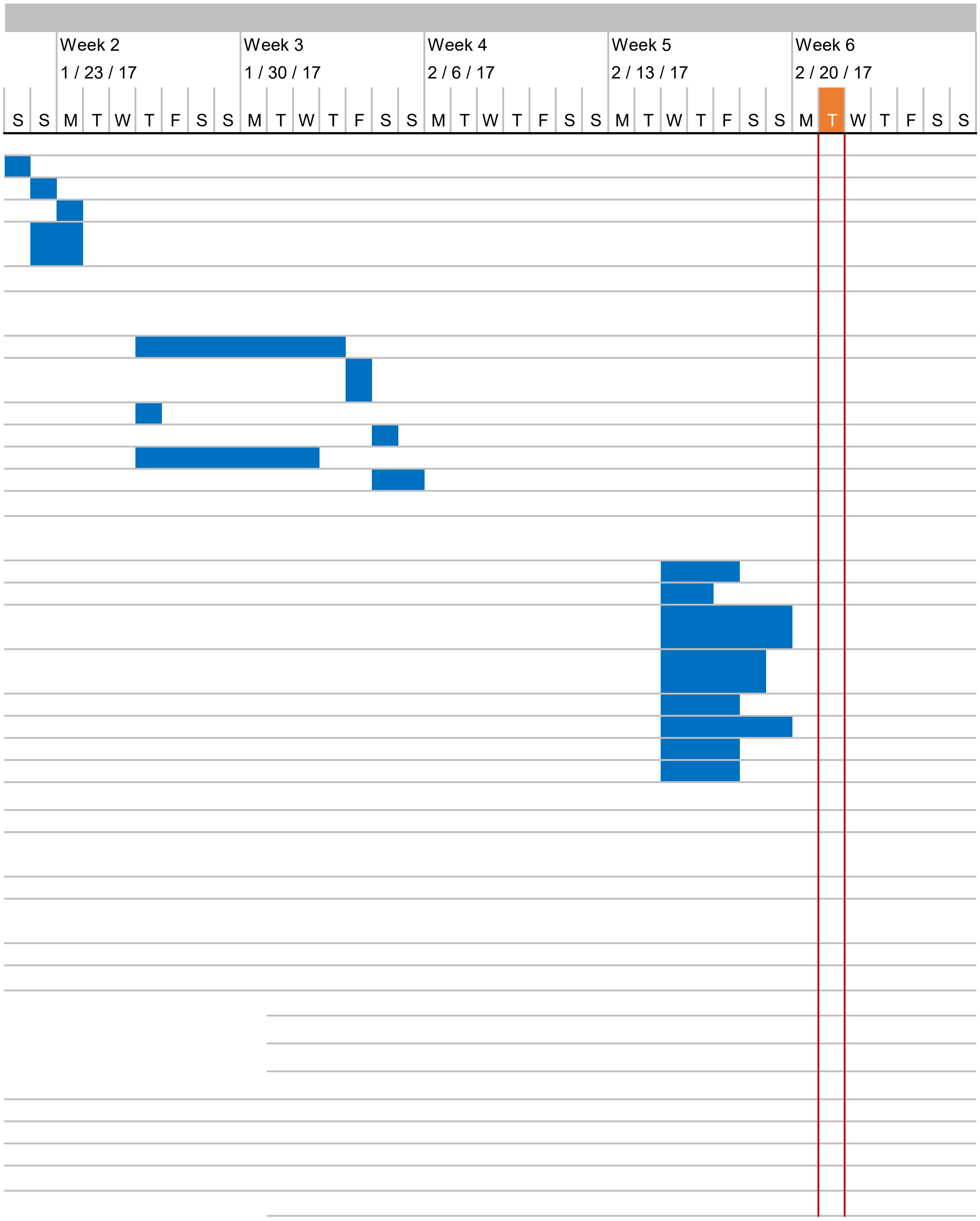
Virtual Stock

Batman

Project Lead:	Mengyuan Zhu
Project Start Date:	1/21/2017 (Saturday)
Display Week:	1

WBS	Task	Lead	Start	End	Cal. Days	% Done	Work Days	Week 1 1 / 16 / 17 M	T	W	T	F
1	**Introduction**											
1.1	Title Page	Mengyuan	Sat 1/21/17	Sat 1/21/17	1	100%	1					
1.2	Resumes	Mengyuan	Sun 1/22/17	Sun 1/22/17	1	100%	1					
1.3	Approved topic	Mengyuan	Mon 1/23/17	Mon 1/23/17	1	100%	1					
1.4	Work Strucutere Document	Mengyuan	Sun 1/22/17	Mon 1/23/17	2	100%	1					
2	**Requirements Elicitation**											
2.1	Problem statement	Mengyaun	Thu 1/26/17	Thu 2/02/17	1	100%	6					
2.2	Requirements Traceabillity Matrix	Mengyaun	Fri 2/03/17	Fri 2/03/17	1	100%	1					
2.3	Gantt Chart	Sungjae	Thu 1/26/17	Thu 1/26/17	1	100%	1					
2.4	Dictionary	Mengyaun	Sat 2/04/17	Sat 2/04/17	1	100%	1					
2.5	Topic rational	Sharon	Thu 1/26/17	Wed 2/01/17	1	100%	1					
2.6	Updated WSD	Mengyaun	Sat 2/04/17	Sun 2/05/17	1	100%	1					
3	**System Analysis & Design**											
3.1	Horizontal prototype	Mengyaun	Wed 2/15/17	Fri 2/17/17	1	100%	3					
3.2	RTM Update	Sungjae	Wed 2/15/17	Thu 2/16/17	1	100%	2					
3.3	Use cases & Interaction Diagram	Mengyaun	Wed 2/15/17	Sun 2/19/17	1	100%	3					
3.4	Function Point Cost Analysis	Sharon	Wed 2/15/17	Sat 2/18/17	1	100%	3					
3.5	Updated WSD	Mengyaun	Wed 2/15/17	Fri 2/17/17	1	100%	3					
3.6	Updated Gantt Chart	Sungjae	Wed 2/15/17	Sun 2/19/17	1	100%	3					
3.7	Dictionary	Hyeun	Wed 2/15/17	Fri 2/17/17	1	100%	3					
3.8	Use cases rational	Sharon	Wed 2/15/17	Fri 2/17/17	1	100%	3					
4	**Object Design**											
4.1	Software architecture used				1	0%	0					
4.2	RTM Update	Jakub			1	0%	0					
4.3	Category Interaction Diagram				1	0%	0					
4.4	Design of objects				1	0%	0					
4.5	Updated WSD	Mengyaun			1	0%	0					
4.6	Updated Gantt Chart	Sungjae			1	0%	0					
4.7	Dictionary	Hyeun			1	0%	0					
4.8	Object design rational	Sharon			1	0%	0					
5	**Rationale**											
5.1	Merge rationale	Sharon			1	0%	0					
5.2	RTM Update	Jakub			1	0%	0					
5.3	Updated WSD	Mengyaun			1	0%	0					
5.4	Updated Gantt Chart	Sungjae			1	0%	0					

5.5	Dictionary	Hyeun			1	0%	0
6	**Test Document & Code**						
6.1	RTM Update	Jakub			1	0%	0
6.2	Source code & Executable code				1	0%	0
6.3	Test cases				1	0%	0
6.4	Updated WSD	Mengyaun			1	0%	0
6.5	Updated Gantt Chart	Sungjae			1	0%	0
6.6	Dictionary	Hyeun			1	0%	0
6.7	Test cases rational	Sharon			1	0%	0
6.8	COCOMO						
7	**Final Report**						
7.1	Table of Contents				1	0%	0
7.2	Full documentation				1	0%	0
7.3	Function Point Cost Analysis & COCOMO				1	0%	0
7.4	Prototype				1	0%	0
7.5	Project legacy				1	0%	0
7.6	Final WSD	Mengyaun			1	0%	0
7.7	Final Gantt Chart	Sungjae			1	0%	0
7.8	Dictionary	Hyeun			1	0%	0
7.9	Resumes				1	0%	0
7.10.1	User Guide				1	0%	0
7.10.2	Email from GIT				1	0%	0


Week 2
1 / 23 / 17
Week 3
1 / 30 / 17
Week 4
2 / 6 / 17
Week 5
2 / 13 / 17
Week 6
2 / 20 / 17
S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S

Week 7							Week 8							Week 9							Week 10							Week 11							Week	
2 / 27 / 17							3 / 6 / 17							3 / 13 / 17							3 / 20 / 17							3 / 27 / 17							4 / 3 /	
M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T

12					Week 13						
17					4 / 10 / 17						
W	T	F	S	S	M	T	W	T	F	S	S