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

4/10/2017

Virtual Stock

Contents

Topic Description

We plan to develop a PC software with a user-friendly GUI. The software is for virtual stock trading. The price is real-time while the exchange is virtual. It's easy to use for stock market beginners.

1. Access market quotes and data in real-time
2. Build a personalized stock watch list
3. View detailed charts of historical market data
4. Transfer money from/to a virtual bank
5. View User history

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

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.

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. 3-4 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.

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

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: None
Use Cases Utilized: None
Timing Constraints: None

Use Case 4: Show_Portfolio

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

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.

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).

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.

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

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

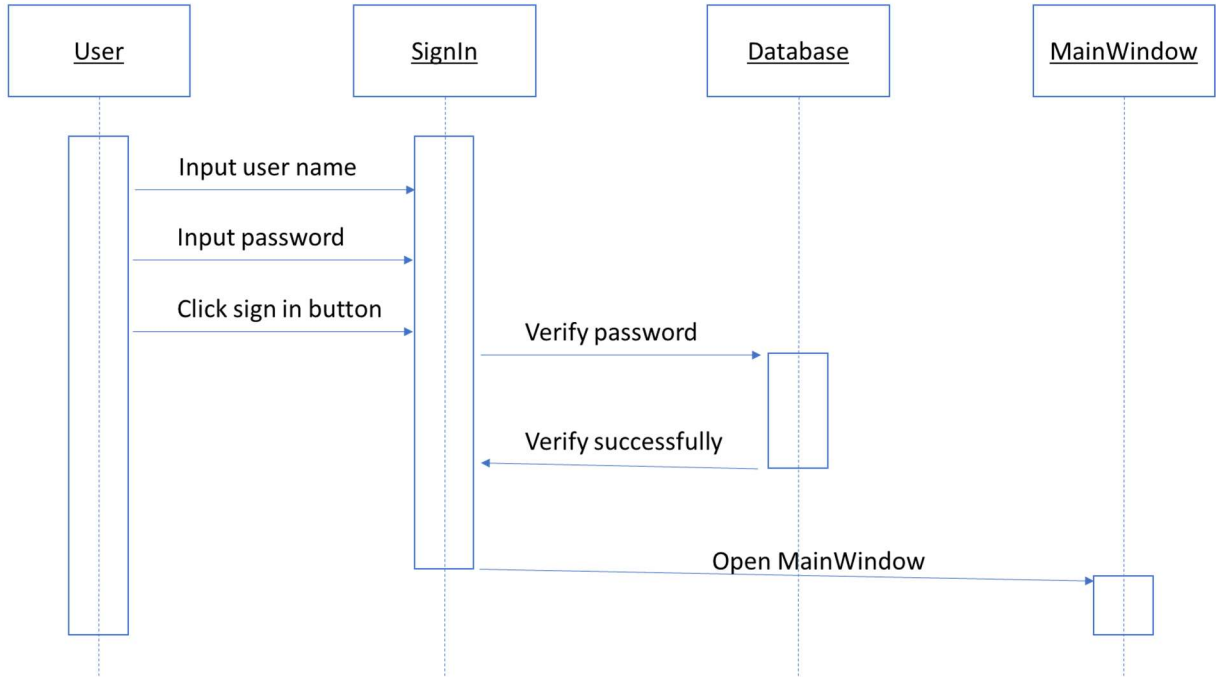
Exceptions:
None

Use Cases Utilized:
Show_Main_Window

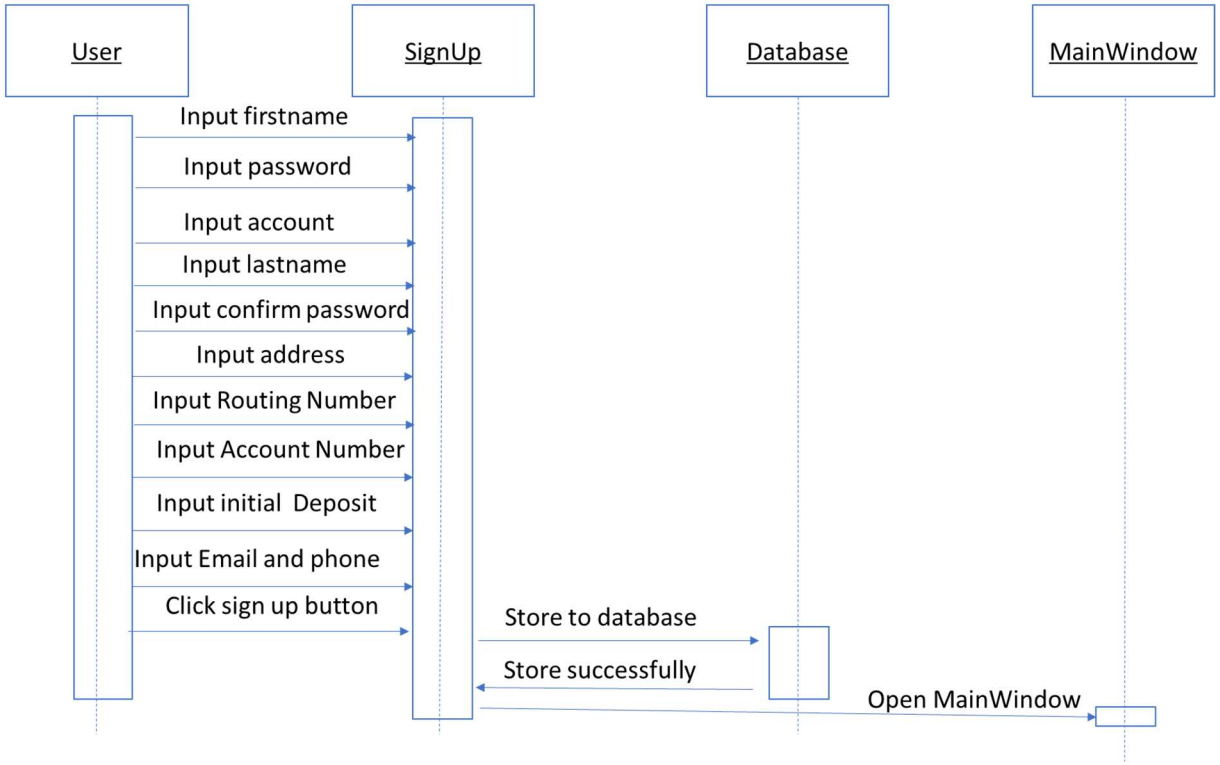
Timing Constraints:
None.

Interaction Diagrams

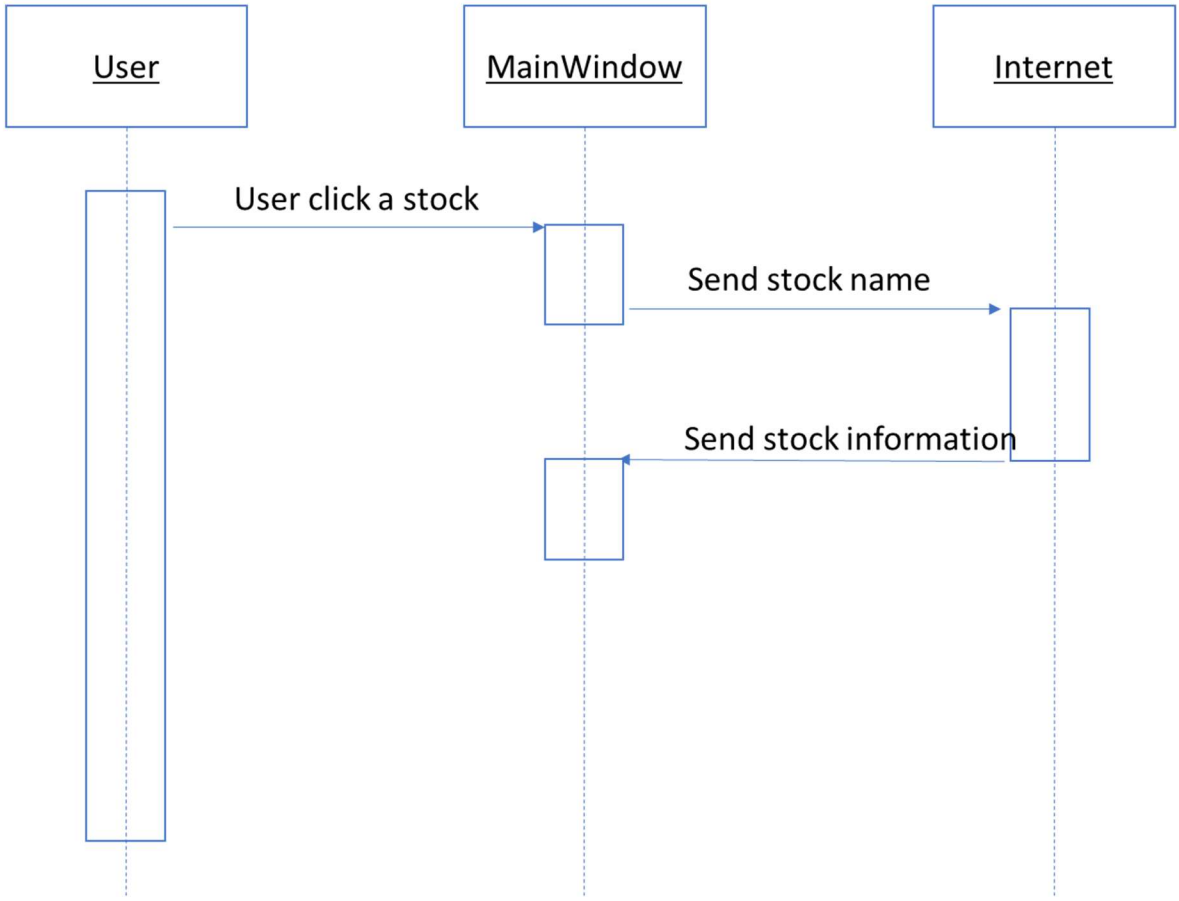
Use Case 1: User_Sign_In



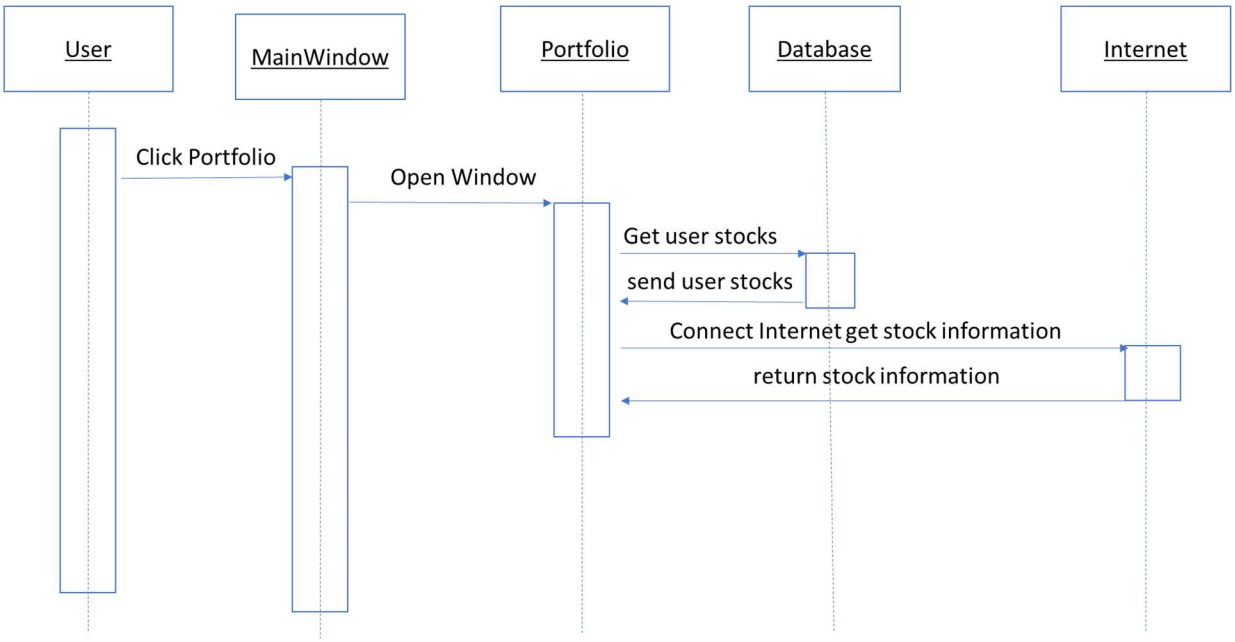
Use Case 2: User_Sign_Up



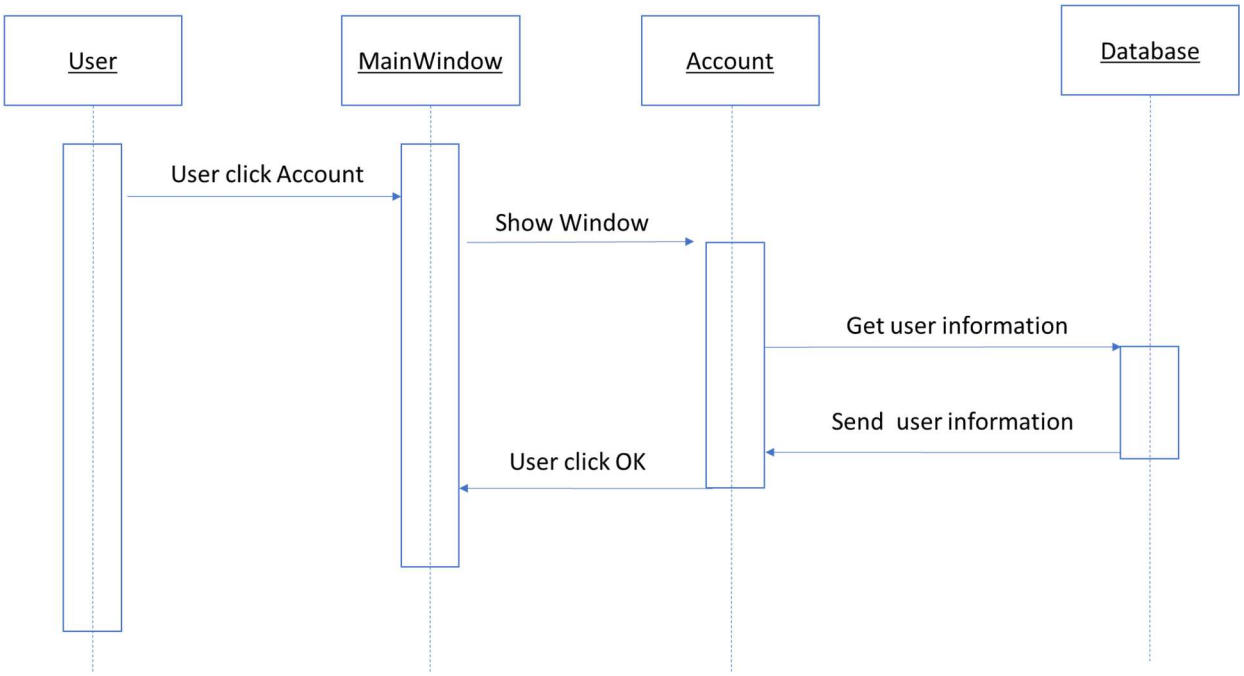
Use Case 3: Show_Main_Window



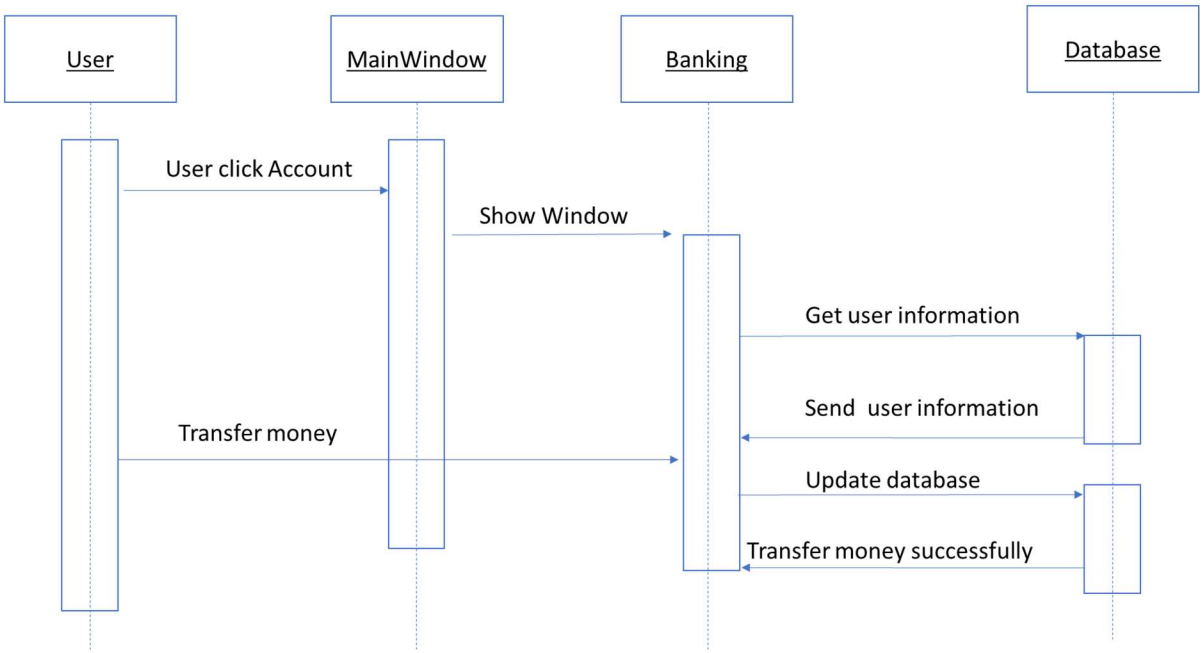
Use Case 4: Show_Portfolio



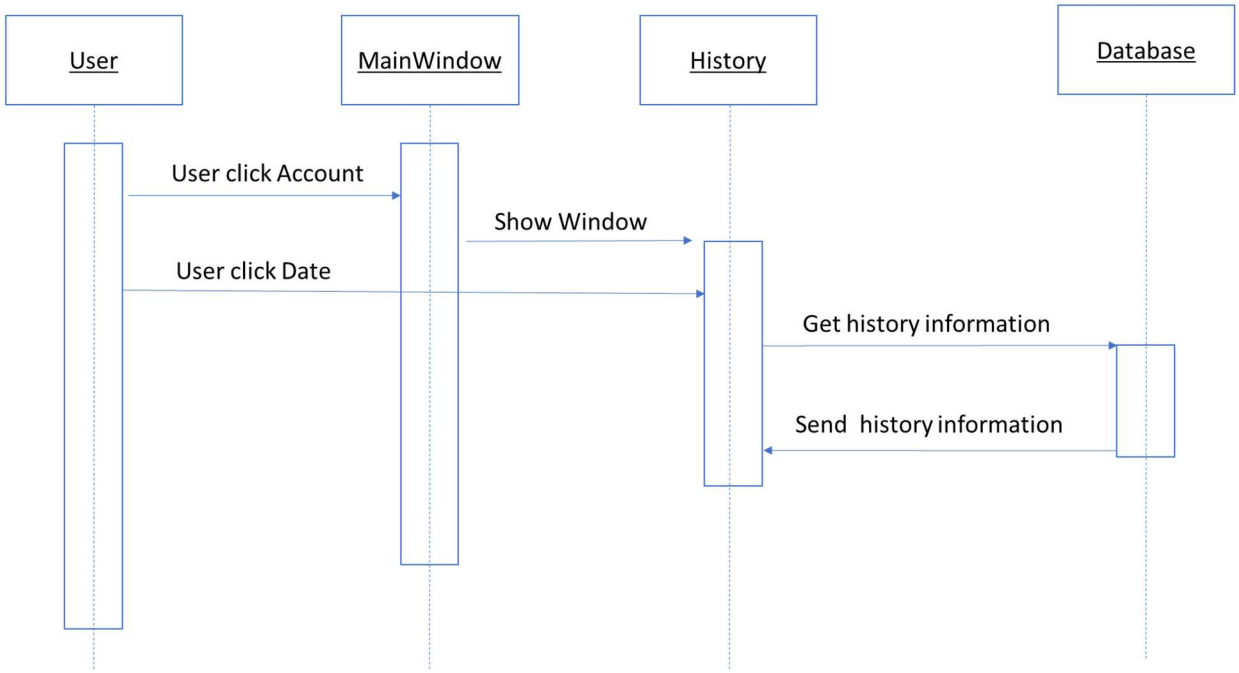
Use Case 5: Show_Account



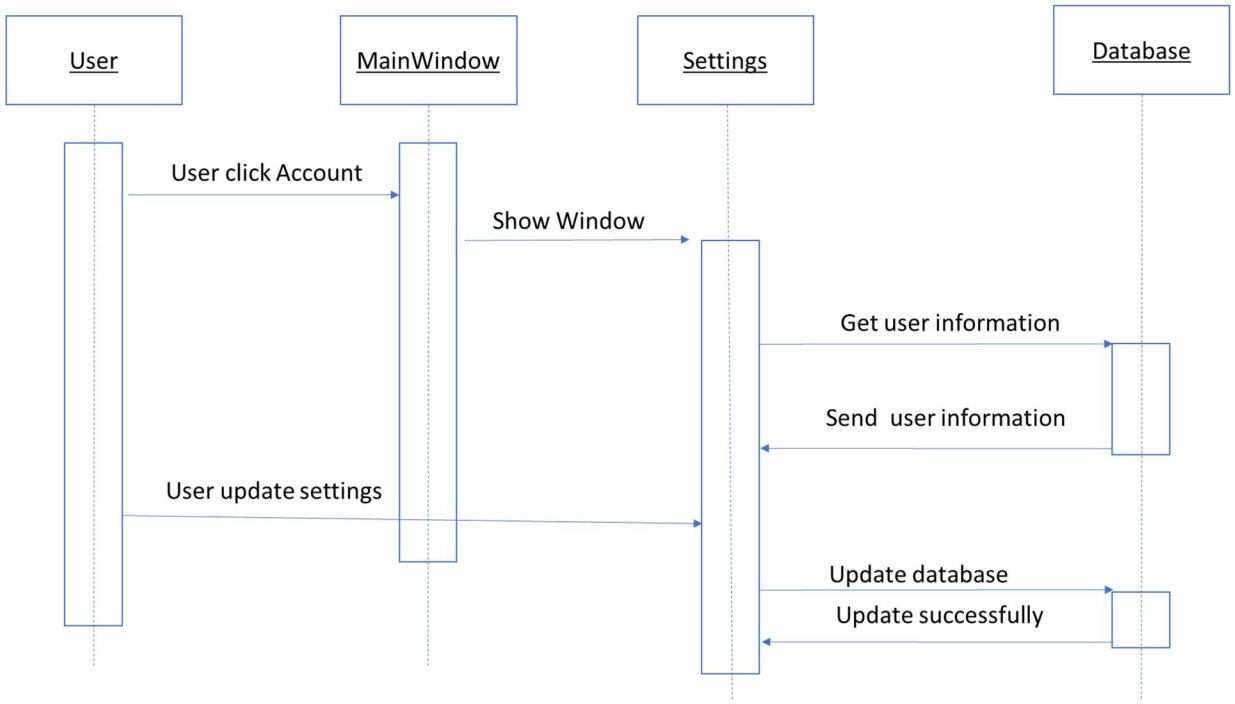
Use Case 6: Show_Banking



Use Case 7: Show_History



Use Case 8 Show_Settings



Software Architecture

Virtual Stock software is in the Model/View/Controller (MVC) architecture style.

Models: Stock information and user information

Views: SignIn, SignUp, MainApp, History, Account, Portfolio, Banking and Settings

Controllers: controller is used to start the views and process information of stocks and users.

Database

SQLite

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 Rational Architecture
Sungjae Kim	Finalize code documentation Java coder Requirements Traceability Matrix Gannt Chart Category Interaction Diagram
Sharon Kim	User Guide Function Point Cost Analysis. Program tester Rational
Jakub Pietrasik	Rational
Hyeun Kang	Object Design

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.

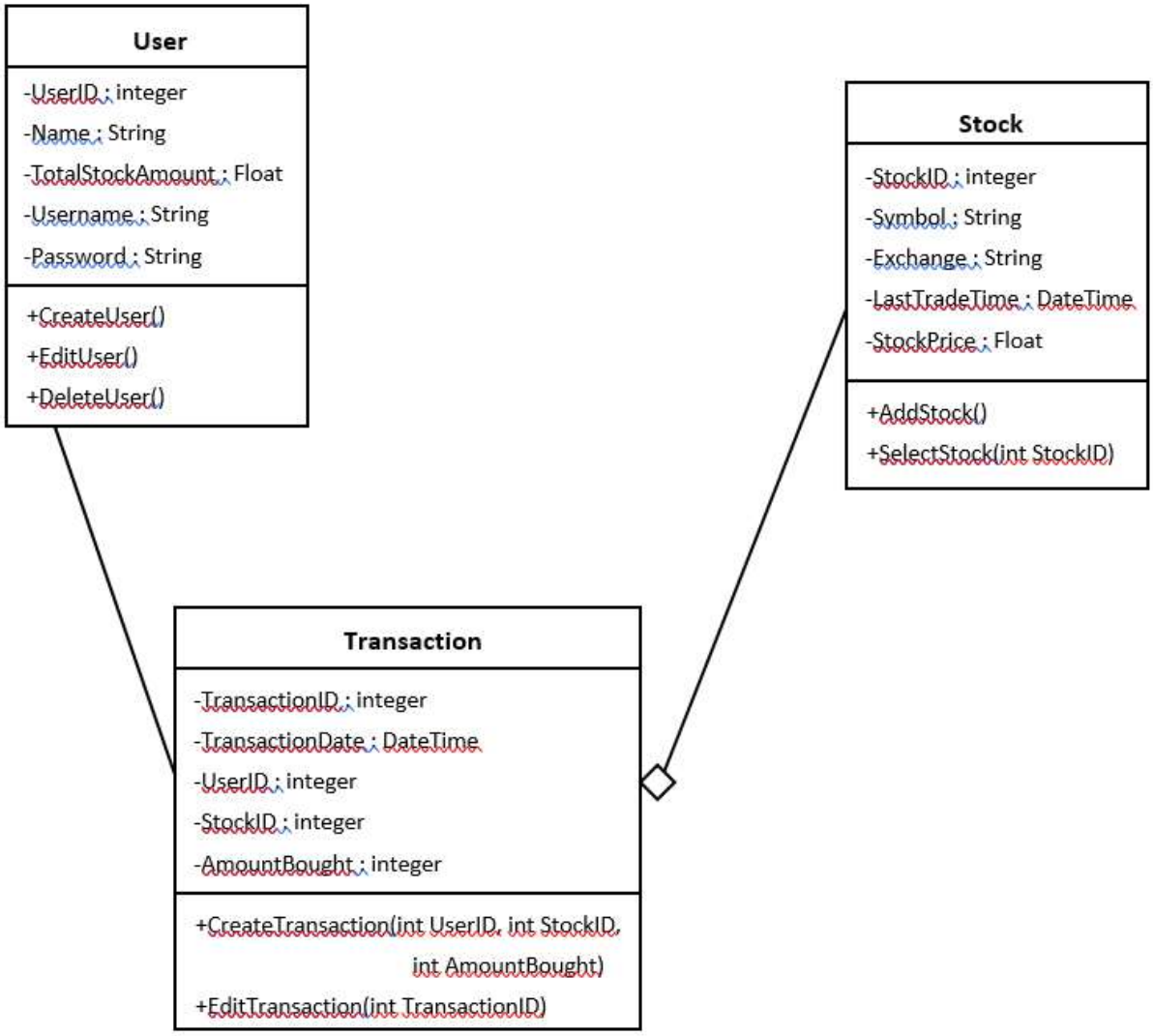
GUI: The graphical user interface is a type of user interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, instead of text-based user interfaces, typed command labels or text navigation.

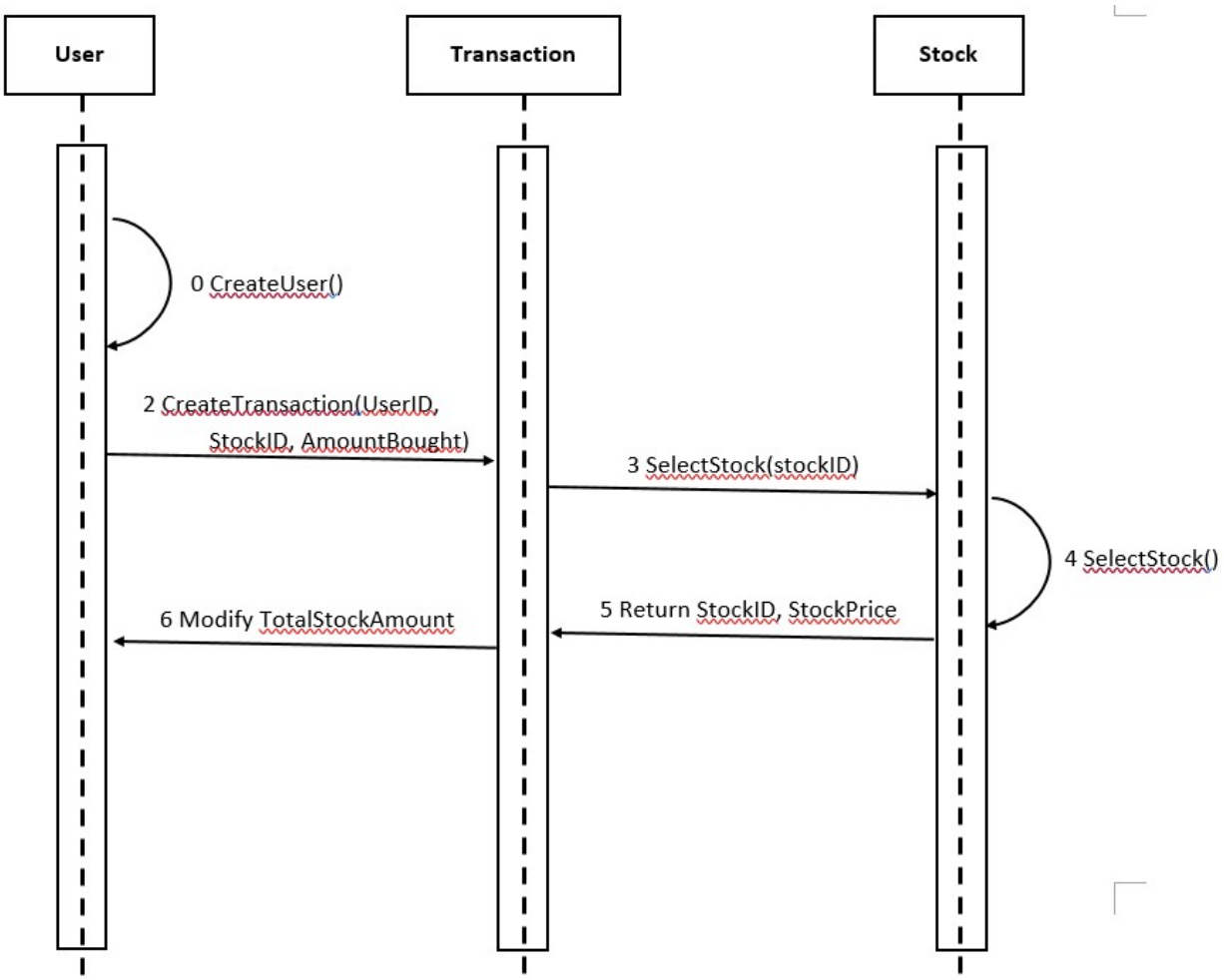
HTTP: The Hypertext Transfer Protocol (HTTP) is an application protocol for distributed, collaborative, and hypermedia information systems. HTTP is the foundation of data communication for the World Wide Web.

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.

Object Design

Category Interaction Diagram


User
-UserID : integer
-Name : String
-TotalStockAmount : Float
-Username : String
-Password : String
+CreateUser()
+EditUser()
+DeleteUser()
Stock
-StockID : integer
-Symbol : String
-Exchange : String
-LastTradeTime : DateTime
-StockPrice : Float
+AddStock()
+SelectStock(int StockID)
Transaction
-TransactionID : integer
-TransactionDate : DateTime
-UserID : integer
-StockID : integer
-AmountBought : integer
+CreateTransaction(int UserID, int StockID, int AmountBought)
+EditTransaction(int TransactionID)



Test Cases

ID	1
Title	Sign In
Pre-conditions	User clicks sing in button
Test Steps	1. User types user name 2. User types password 3. User clicks sign in
Expected Results	Account signs in. If password wrong, let user type again

ID	2
Title	Sign Up
Pre-conditions	User clicks sing up button
Test Steps	1. User types user name 2. User types password 3. User types password twice for validation 4. User clicks sign up

Expected Results	Account signs up. If passwords are different, let user type again

ID	3
Title	Buy stock
Pre-conditions	User clicks the buy button
Test Steps	1.User clicks the buy button
Expected Results	User purchases stocks. If not enough balance in the account, notify user of insufficient balance.

ID	4
Title	Account Edit
Pre-conditions	User changes the account information
Test Steps	1. User types new name. 2. User types new email. 3. User types new password. 4. Click submit button.
Expected Results	Account updated. If new password does not match confirm password, notify the user of unmatched password.

ID	5
Title	Search Stock
Pre-conditions	User types in the symbol for stock
Test Steps	1. User types the symbol for stock 2. User clicks search button
Expected Results	New window with detailed stock information pops up. If wrong symbol name, notify the user of invalid stock symbol.

Rational

Batman is a software for virtual stock trading where the price is real-time while the exchange is virtual. Users can access market quotes and data in real time, build a personalized stock watch list, view detailed charts of historical market data, transfer money from/to a virtual bank, and view user history. The database used in this software is portfolio.db, which is read with an SQLite library.

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, user sign up and user

sign in to Batman ensures that a user has a Virtual Stock account before using Virtual Stock's main functions.

In Batman, users have the access to the main window, portfolio, account, bank, history, and settings. 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. By showing portfolio, users have access to see days, weeks, months, and years' worth of stocks. By showing account, users have easy access to their money and funds which they can withdraw from their virtual stock to their bank. By showing banking, this 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). By showing history, users can see specific dates of stocks and their prices of when they are bought or sold. 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.

The objects in Batman are:

Stock.java
Stores information pertaining to stock attributes. Includes name, price, shares, open, TodayHigh, TodayLow, YearHigh, YearLow, Volume, Marketcap, PARatio, and DivyYield values.

StockDayData
Manages retrieval of the stock data from Yahoo and contains method for getting the URL source.

StockListWrapper
Wrapper for Stock List that has a getter and a setter for the person object.

User
Stores information about the user including attributes such as name, address, social security number, and date of birth.

DateUtil
Finds and formats the date from dateString.

AccountController
Provides the elements of the visual interface for the Account screen. This includes labels for total, stock and cash, as well as an OK and a withdraw button.

BankingController
Provides the elements of the visual interface for the Banking screen. This includes buttons for transferring between the bank and VS, automatic depositing, and linking accounts.

HistoryController
Provides the elements of the visual interface for the History screen. This includes a combo box for the date range and a table of the history.

PortfolioController
Provides the elements of the visual interface for the Portfolio screen. This includes elements such as a bar chart, buttons for managing time intervals for day/week/month, and allows for the setting of person data.

RootLayoutController
Provides the elements of the visual interface for the Root Layout screen. Handles the Sign In, Sign Up and Showing of portfolio, banking, account and history screens.

SettingController
Provides the elements of the visual interface for the Setting screen. This includes text fields for account name and password, email, phone, address as well as the buttons for updating the content with that entered as well as one for logging out.

SignInController
Provides the elements of the visual interface for the Sign In screen. Includes buttons for signing in, registering and text fields to allow for the entering of a username and a password.
SignUpController
Provides the elements of the visual interface for the Sign Up screen. Includes labels for the name, password, account name, address, routing number, account number, initial deposit, phone and email contact information. Also includes a button to confirm and one to cancel.
StatisticsController
Provides the elements of the visual interface for the Statistics screen. Includes a bar chart and category axis of stock information. Allows for the viewing of and setting of Person Data.
StockDayController
Provides the elements of the visual interface for the Stock Day screen. Allows for the setting of stock quote data as well as the controller for Stock Day.
StockOverviewController
Provides the elements of the visual interface for the Stock Overview screen. Manages overview elements including stock name, stock price, shares, open, today high, today low, year high, year low, volume, average volume, market cap, PERatio, and div field. Shows person details.
StockPurchaseController
Provides the elements of the visual interface for the Stock Purchase screen. Displays the elements of stock name, shares, and the dialog box that appear when making a stock purchase.

The Virtual Stock software is in the Model/View/Controller (MVC) architecture style. Subsystems are classified into three different type. It has these reasons: Simultaneous Development - Multiple developers can work simultaneously on the model, controller and views. High Cohesion - MVC enables logical grouping of related actions on a controller together. The views for a specific model are also grouped together. Low Coupling - The very nature of the MVC framework is such that there is low coupling among models, views or controllers. Ease of modification - Because of the separation of responsibilities, future development or modification is easier. Multiple views for a model - Models can have multiple views.

The test cases in Batman are functional to make sure user can use it. When Batman executes test cases, it tells us about the functionality of the application and talks about the desired output to be seen.

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

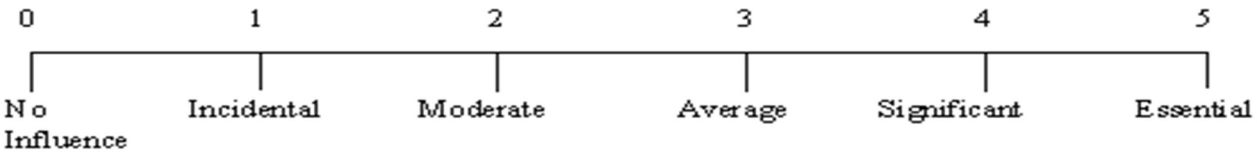
1	x	○ 5	◉ 7	○ 10	=	7

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

Reset

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

Construction Cost Model

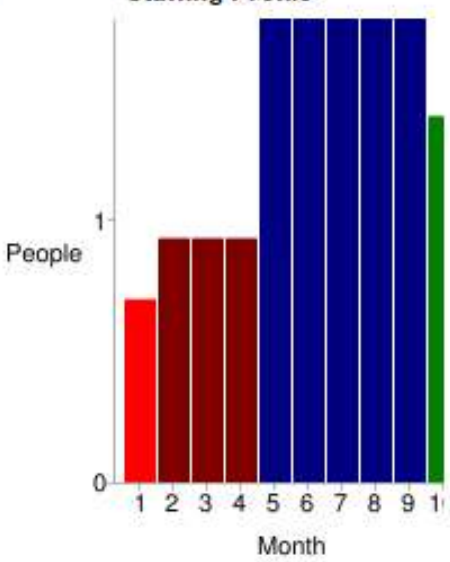
Results

Software Development (Elaboration and Construction)

Effort = 12.1 Person-months
Schedule = 8.4 Months
Cost = $12125

Total Equivalent Size = 5000 SLOC

Staffing Profile



Acquisition Phase Distribution

Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	0.7	1.0	0.7	$728
Elaboration	2.9	3.1	0.9	$2910
Construction	9.2	5.2	1.8	$9216
Transition	1.5	1.0	1.4	$1455

Software Effort Distribution for RUP/MBASE (Person-Months)

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.1	0.3	0.9	0.2
Environment/CM	0.1	0.2	0.5	0.1
Requirements	0.3	0.5	0.7	0.1
Design	0.1	1.0	1.5	0.1
Implementation	0.1	0.4	3.1	0.3
Assessment	0.1	0.3	2.2	0.3
Deployment	0.0	0.1	0.3	0.4

Conclusion: Project has 278 FP and the cost $12125

Source Code

https://github.com/MengyuanZhu/SoftwareEngineering

Resumes

SUNGJAE KIM

605 James Lee Dr. Suwanee, GA 30024

404.775.6111 | sungjkim.com | sungjkim34@gmail.com

Summary

Motivated student seeking internship and employment as part of a dynamic software development team. Looking for the job opportunities to work as a software developer not only where I can contribute my skills and abilities to the growth of the organization but also where I can build my professional career and gain knowledge and expertise.

Education

Pursuing Bachelor's Degree, Computer Science 2013 – May 2018

Georgia State University | *Atlanta, GA*

Highlights

Object-Oriented | Java, C#, C (basic)

Web Design | HTML, CSS, JavaScript, Bootstrap Framework

CMS | WordPress

Back-End Development | MySQL, PHP, Node.js (basic)

Hardware-Oriented | Arduino C# programming (basic), Assembly (basic)

Other Knowledge | Microsoft Office (Words, PowerPoint)

Operating System | Linux/UNIX, Microsoft Windows, Mac OS

Experience

Software Development Co-Op 2017-Present

UPS | *Atlanta, GA*

Work with the Transportation Technology Group (TTG) to develop geofencing and beacons to use for inventory tracking at the Atlanta corporate office.

Software & Web Development Intern 2016 -Present

Fyminds| *Atlanta, GA*

MyICHot - Set up and maintain sites from start to finish for clients.

LifeArtCabinetry - Designed a popup based off jQuery.

Researched to build plugin for importing XML file from WordPress e-commerce site into an application.

TireAccents – Set up WordPress site for a startup tires decal company.

Worktive - Android Java development. Use of custom API to search for jobs. Android App Development Use of Java, JSON, Parser, MySQL, etc.

Built java program to scrap from indeed.com demonstrates use of Jaunt and regex pattern/matcher.

PantherHackers Tech Committee 2016 - Present

Georgia State University | *Atlanta, GA*

Hand-coded kata website to join the PH Tech Committee.

Built and uploaded local dev environment to GitHub for their current website

Math Instructor 2015 - 2016

Mathnasium | *Norcross, GA*

Tutor students K-12 in school math and SAT math.

1st place in center instructors' math equation competition.

Languages

Fluent in Korean & English

Sharon Kim
818 Burns Estates Drive • Lilburn, GA 30047 • 404-998-6187 • sharkim1104@gmail.com

---

RELATED CLASSES

· Principles of Computer Science · Principles of Computer Programming · Theoretical Foundations of Computer Science, · System Level Programming · Data Structures · Computer Org & Programming · Computer Architecture · Programming Language Concepts · Intro to Compilers · Software-Engineering

EDUCATION

Georgia State University Atlanta, GA
Bachelor of Science in Computer Science May 2018

SKILLS

Programming Language: Java, C, C++, Assembly
Applications: Proficient in MS Word, MS Excel, MS PowerPoint
Operating System: Microsoft Windows, Linux/UNIX
Language: Intermediate written and conversational Korean

EXPERIENCE

Collins Hill Leadership Team 2010-2014
Member
· Built strong connections with younger peers through tutoring, counseling, and volunteering
· Offered supportive guidance to struggling peers

Future Business Leadership of America
Leader 2010-2014
· Encouraged other students to search for their interest in the business enterprise
· Developed and maintained relationships with peers to discuss different issues

State Science Fair
3rd Place 2011
· Placed 3rd place in science fair through researching upon whether student's usage of cellular devices had a correlation with grades

Key Club
Member 2012-2013
· Volunteered through hands-on service to build the school and community
· Developed leadership skills to initiate transforming communities

Korean First Presbyterian Church
Teacher/Volunteer of Vacation Bible School 2012-2016
· Worked closely with parents, pastors, and 20+ volunteer staff to plan events including crafts, music, recreation for 50+ children
· Taught 1st graders Sunday School during a week-long camp

SOFTWARE ENGINEER INTERN

Hyeun Kang

Contact

Email: 7hkang7@gmail.com

Phone#: 678-978-5687

Address

3412 James Harbor Way

Lawrenceville, GA 30044

Objective

Enthusiastic computer science student searching for internship position

Improve my programming skill outside school

Learn to communicate and work with others as a team

Education

Georgia State University – Bachelor of Science in Software Development, expected July 2017

Related Courses

- Data Structure
- Computer Architecture
- Windows System Programming
- Programming Langue Concept
- Software-Engineering
- Compiler Design
- Network

Skills

.NET, C, JAVA

Experience

Computer Technician at Asiana Telecom, Duluth GA 2013-2016

- Repair computers and laptops
- Install network for companies

Private Tutor for SAT Math 2009-2012

Language

English – fluent

Korean – native

Jakub Pietrasik
jpietrasik1@student.gsu.edu

COMPUTER PROGRAMMER

- Student of GSU University's BS in Computer Science program.
- Consistently commended by professors and internship supervisor for programming abilities, grasp of multiple technologies and attention to detail.

- Strong knowledge of object-oriented programing and application development tools using Microsoft VB.Net, C#, .Net, C++, ASP.Net; Python and VBA.
- Known as a self-starter, team player and multitasker--strive to consistently exceed expectations.

Education

Georgia State University, Atlanta, GA
Bachelor of Science, Major in Computer Science, Currently Enrolled

Course Highlights:

- CSC 4110 Introduction to Embedded Systems Laboratory (4)
- CSC 4310 Parallel and Distributed Computing (4)
- CSC 4320 Operating Systems (4)
- CSC 4340 Introduction to Compilers (4)
- CSC 4360 Network-Oriented Software Development (4)
- CSC 4370 Web Programming (4)
- CSC 4380 Windowing Systems Programming (4)

Technology Summary

- Programming/Languages: VB.Net; C#; .Net; C; C++; ASP.Net; Python, VBA, Java, Visual Basic; SharePoint; PHP; MySQL; HTML; Ant
- Databases Management: Oracle 8.x/9.x, SQL Server, MS Access
- Design & IDE Tools: Rational Rose, UML, WSAD, Oracle WebLogic Server
- Systems: Windows Server 2016, Linux/Unix, Mac OS X

User Guide

1. Signing-up
   - Run the Virtual Stock application
   - Click Sign-Up
   - Enter first name, last name, username, email, and password
   - Click Sign-Up
2. Signing-in
   - Run the Virtual Stock application
   - Enter in Username and Password
   - Click Sign-in

3. Details of Stocks
   - Click on the stock name to see details of the stocks' price, shares, open, volume, today's high, average volume today's low, market capacity, 52 week high, 52 week low, P/E ratio and Diy/Yield
4. Buying shares
   - Click buy and then a window will show the price
   - Enter the number of shares to be purchased
5. Selling shares
   - Click sell
   - Enter the number of shares to sell
6. Portfolio
   - Click portfolio
   - Enter stock name
   - Click search
7. Withdrawing funds
   - Click account
   - Click withdraw funds
   - Enter the amount of funds to withdraw
8. Transferring money

   I. To Virtual stock
   - Click banking under account
   - Click transfer to Virtual stock
   - Select the bank account and money to transfer

   II. To Bank
   - Click banking under account
   - Click transfer to bank
   - Select bank account and money to transfer

   III. Automatic deposits
   - Click banking under account
   - Click automatic deposits
   - Select the time and money to be transferred from Bank account to Virtual stocks

   IV. Link account
   - Click banking under account
   - Click link account
   - Select a bank account and enter the routing number and account number
9. View History
   - Click account
   - Click history
10. Edit Account
    - Click on Account
    - Can edit account name, email address, password, phone number, and address

Gantt Chart

Virtual Stock

Batman

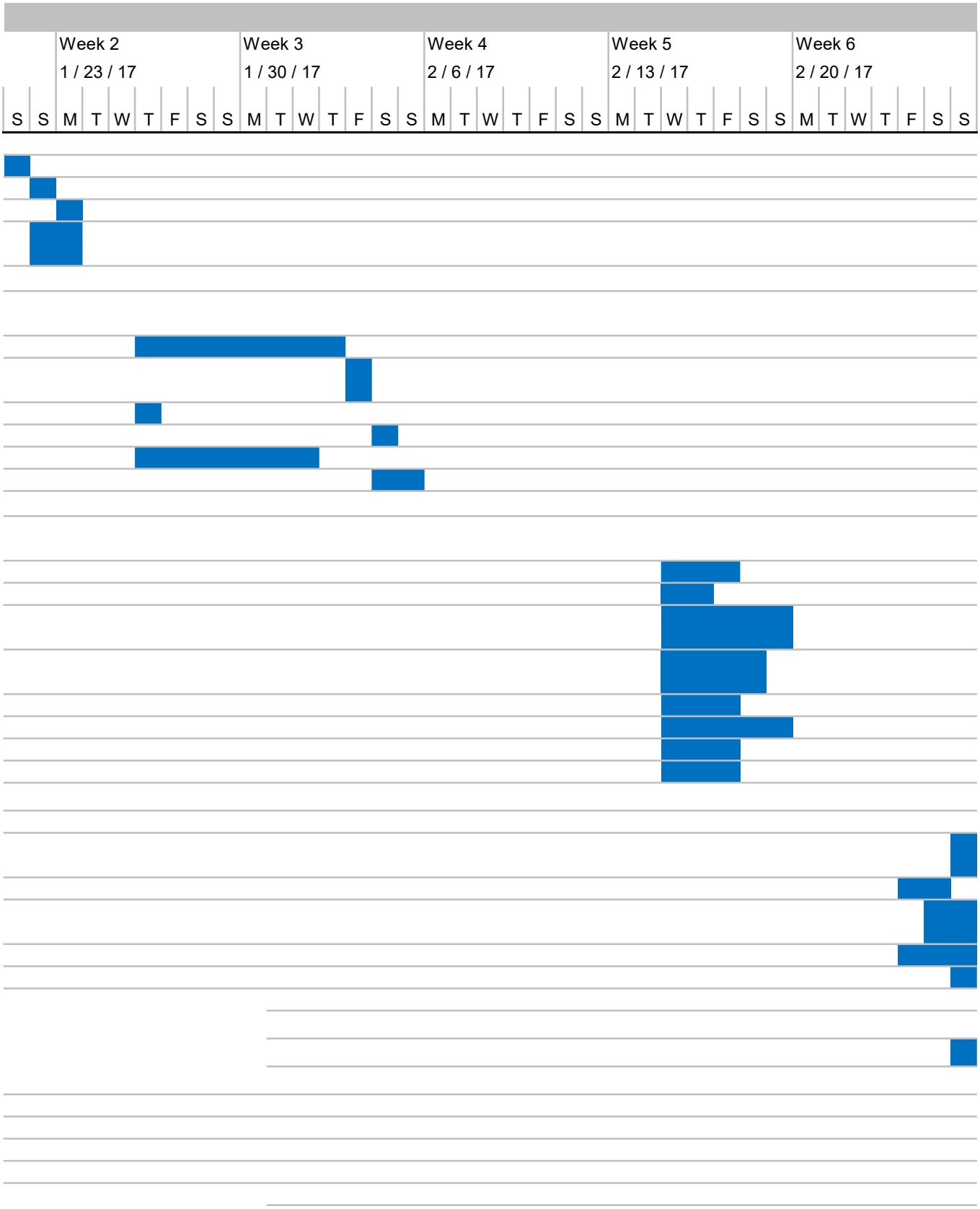
Project Lead: Mengyuan Zhu

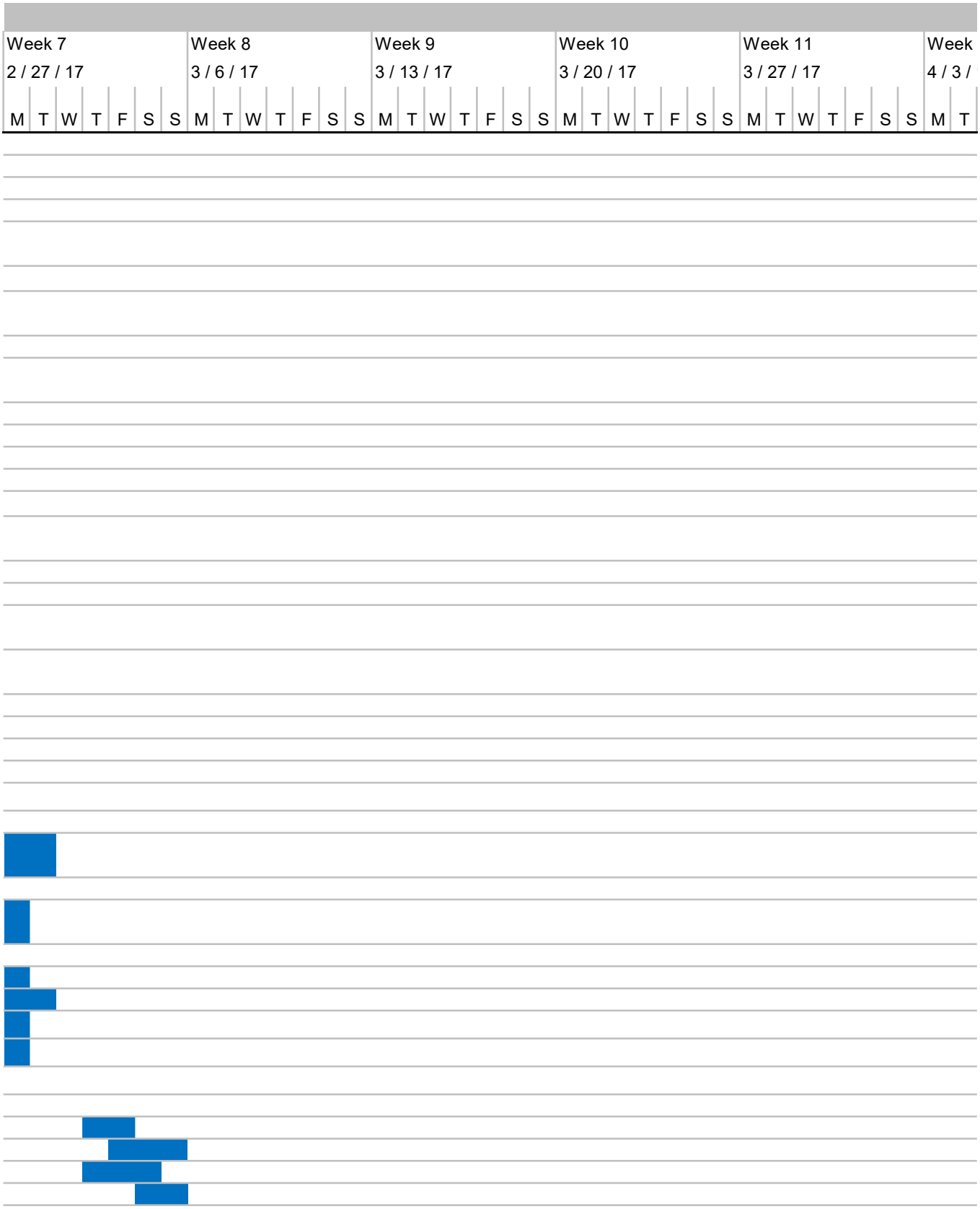
Project Start Date: 1/21/2017 (Saturday)

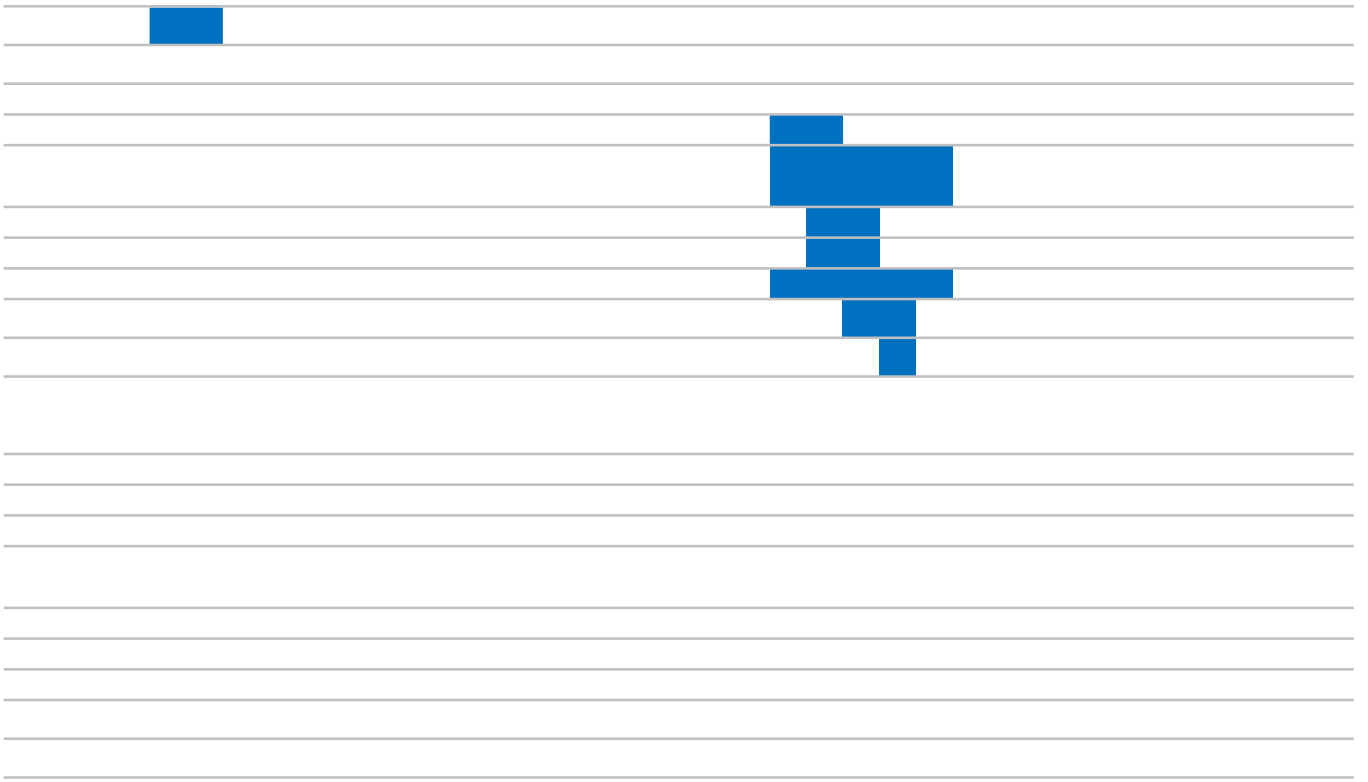
Display Week: 1

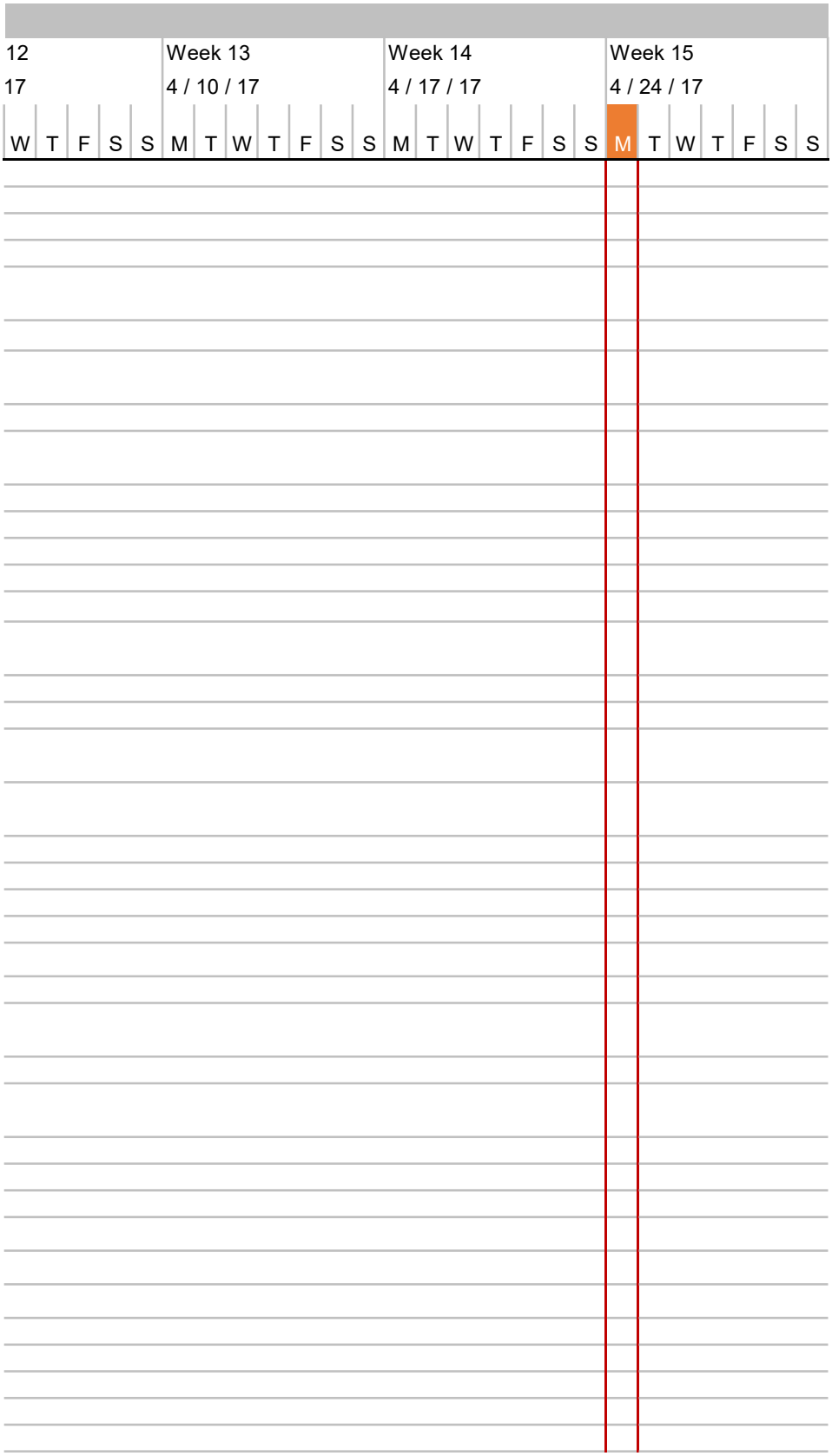
WBS	Task	Lead	Start	End	Cal. Days	% Done	Work Days	Week 1 1 / 16 / 17 M	T	W	T	F
1	**Introduction**											
1.1	Title Page	Mengyuan	Sat 1/21/17	Sat 1/21/17	1	100%	1					
1.2	Resumes	Mengyuan	Sun 1/22/17	Sun 1/22/17	1	100%	1					
1.3	Approved topic	Mengyuan	Mon 1/23/17	Mon 1/23/17	1	100%	1					
1.4	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	Hyeun	Sun 2/26/17	Tue 2/28/17	2	100%	2					
4.2	RTM Update	Mengyaun	Fri 2/24/17	Sat 2/25/17	1	100%	1					
4.3	Category Interaction Diagram	Sungjae	Sat 2/25/17	Mon 2/27/17	2	100%	1					
4.4	Design of objects	Jakub	Fri 2/24/17	Sun 2/26/17	2	100%	1					
4.5	Updated WSD	Mengyaun	Sun 2/26/17	Mon 2/27/17	1	100%	1					
4.6	Updated Gantt Chart	Sungjae	Mon 2/27/17	Tue 2/28/17	1	100%	2					
4.7	Dictionary	Mengyaun	Mon 2/27/17	Mon 2/27/17	1	100%	1					
4.8	Object design rational	Sharon	Sun 2/26/17	Mon 2/27/17	2	100%	1					
5	**Rationale**											
5.1	Merge rationale	Sharon	Thu 3/02/17	Fri 3/03/17	1	100%	1					
5.2	RTM Update	Jakub	Fri 3/03/17	Sun 3/05/17	1	100%	2					
5.3	Updated WSD	Mengyaun	Thu 3/02/17	Sat 3/04/17	1	100%	2					
5.4	Updated Gantt Chart	Sungjae	Sat 3/04/17	Sun 3/05/17	1	100%	1					

5.5	Dictionary	Hyeun	Fri 3/03/17	Sat 3/04/17	1	100%	1
6	**Test Document & Code**						
6.1	RTM Update	Mengyaun	Mon 3/20/17	Tue 3/21/17	2	100%	2
6.2	Source code & Executable code	Sungjae	Mon 3/20/17	Fri 3/24/17	4	100%	5
6.3	Test cases	Mengyaun	Tue 3/21/17	Wed 3/22/17	2	100%	2
6.4	Updated WSD	Mengyaun	Tue 3/21/17	Wed 3/22/17	1	100%	2
6.5	Updated Gantt Chart	Sungjae	Mon 3/20/17	Fri 3/24/17	1	100%	5
6.6	Dictionary	Hyeun	Wed 3/22/17	Thu 3/23/17	1	100%	2
6.7	Test cases rational	Sharon	Thu 3/23/17	Thu 3/23/17	1	100%	1
6.8	COCOMO	Mengyaun	Wed 3/22/17	Thu 3/23/17	2	100%	2
7	**Final Report**						
7.1	Table of Contents	Mengyaun	Thu 4/20/17	Sun 4/23/17	2	100%	2
7.2	Full documentation	Mengyaun	Thu 4/20/17	Sun 4/23/17	5	100%	2
7.3	Function Point Cost Analysis & COCOMO	Mengyaun	Tue 4/18/17	Thu 4/20/17	1	100%	3
7.4	Prototype	Sungjae	Wed 4/19/17	Mon 4/24/17	6	100%	4
7.5	Project legacy	Jakub	Wed 4/19/17	Sun 4/23/17	1	100%	3
7.6	Final WSD	Mengyaun	Thu 4/20/17	Fri 4/21/17	1	100%	2
7.7	Final Gantt Chart	Sungjae	Thu 4/20/17	Fri 4/21/17	1	100%	2
7.8	Dictionary	Hyeun	Tue 4/18/17	Sun 4/23/17	1	100%	4
7.9	Resumes	Team	Thu 4/20/17	Fri 4/21/17	1	100%	2
7.10.1	User Guide	Mengyaun	Thu 4/20/17	Sun 4/23/17	2	100%	2
7.10.2	Email from GIT	Team	Thu 4/20/17	Fri 4/21/17	1	100%	2


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
2 / 27 / 17
Week 8
3 / 6 / 17
Week 9
3 / 13 / 17
Week 10
3 / 20 / 17
Week 11
3 / 27 / 17
Week
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
17
W T F S S
Week 13
4 / 10 / 17
M T W T F S S
Week 14
4 / 17 / 17
M T W T F S S
Week 15
4 / 24 / 17
M T W T F S S