# ECS414U/A Miniproject form

Queen Mary University of London

2021/22

|  |  |
| --- | --- |
| Name | Molla Fahad Kolim |
| Student ID | 200438670 |
| Submitted file name | Investment App |
| Level of this program (1, 2, 3, Extra) | Completed up to extra |

|  |
| --- |
| Brief description of the program. Write the chosen theme and a high-level overview of the features (two or three sentences should suffice). |
| Created an investment platform that allows the user to view their balance, deposit and withdraw money, buy/sell stocks whose price and quantity values are randomly changing every time the application is ran, view their portfolio as well as load/save their data for later use. |

|  |  |
| --- | --- |
| List all your source code files, and briefly describe their roles. Add as many rows as necessary. Mark the main file used for compilation in bold. | |
| File name | Description |
| AMD | A subclass of parent class ‘Stock’ – holds information such as the name, randomly generated price, randomly generated quantity of shares, a new price for the share for when it’s time to sell and a string of information which is later used. |
| Apple | A subclass of parent class ‘Stock’ – holds information such as the name, randomly generated price, randomly generated quantity of shares, a new price for the share for when it’s time to sell and a string of information which is later used. |
| Nvidia | A subclass of parent class ‘Stock’ – holds information such as the name, randomly generated price, randomly generated quantity of shares, a new price for the share for when it’s time to sell and a string of information which is later used. |
| Stocks | A superclass from which AMD, Apple and Nvidia inherited from. This class initializes the variables, as well as setting some ‘ground rules’ such as the range of what the randomly generated price and quantity of shares will be within, as well as getters and setters for various private variables. |
| Transition | This class oversees hiding and showing different scenes of the app, when application is ran the text box, buttons and such are hidden from the users view only showing them the title and start/load button. After user chooses their option, they are transitioned to the menu of the app where the start/load buttons are hidden, and the text area and other buttons are shown |
| User | This class oversees the user’s balance, it holds getters and setters for the user’s private balance variable and deposit/withdraw methods. Unless data is loaded from previous sessions the users balance starts at 0. |
| UserInterface | This class oversees the whole applications GUI, using swing we can alter every different aspect of the GUI. I started by splitting the screen up in 3 sections – 1 being the title area, 1 being the text area where text is going to be outputted and last panel being the buttons area where the user will have a choice to choose their next action. I also specified the background colour, the foreground colour, the size of screen, size of different elements etc. |
| ButtonsActions | This class oversees the working of the 9 buttons that have been created, if a button is pressed then they are redirected to their corresponding method – holds many methods that all do different tasks such as deposit money or sell stocks |
| **InvestmentApp** | This is the main class which calls all the methods together to make the application work, initializes the different classes and calls them whenever the application is ran. |

|  |
| --- |
| Class diagram, in the format specified in the instructions. |
|  |

|  |
| --- |
| Usage instructions. Describe briefly what features are available to the user and how to use them. If File I/O is used, list and describe the files involved. |
| When the application is first started the user is shown the app title, creator title and 2 buttons called Start and Load. If load button is clicked, then userdetails.txt file is read – if it doesn’t exist then they are asked to input their name from which they will be redirected to main page – if file does exist, they will be redirected to main page with their previous data loaded onto the application. If name is clicked, then they are asked to input their name – from which they will be redirected to main page.  Main page consists of text area where output will be shown to user, 9 buttons with varying actions, title and app close button. The buttons range from allowing the user to deposit/withdraw funds from their bank, view their balance, view stock price – keeps changing every time a new session is made – buy and sell stocks, view their portfolio as well as save their data – data is saved by creating a new file called userdetails.txt.  Deposit function allows the user to deposit money into account – if decimal or string is inputted instead of integer then an error message will be printed  Withdraw function allows user to take money out their account if there’s any at all – cannot overdraw from account else transaction will be cancelled and an error message will be outputted, also doesn’t allow any decimals or strings to be inputted  Home button takes the user back to home screen where details of how app works is displayed  Show funds shows the user’s current balance  Show stock displays one of the 3 stocks and their price/quantity and bit of information about it  Buy/Sell stock allows you to buy/sell your stocks if you own any at all at changing prices (new prices are generated every time a new session is made)  Show portfolio shows the user how much of each stock they own  Save button saves the users details, balance, stocks by creating a new file called userdetails.txt  Load button reads the saved data is file userdetails.txt and loads it into the application  When save button is pressed a new .txt file is created called userdetails.txt in Investment App folder, this stores the users’ details. This is also the same file which data is read from when the load button is pressed.  How to use:  Deposit some money, view some stocks, buy some stocks, look at portfolio, sell some shares, view your funds, withdraw money from account/save details. Come back another time so prices can change. |

|  |
| --- |
| Other comments. |
| (Example of me compiling the program on my own machine where my Investment App folder is at my desktop)  As this project uses a package structure in order to run follow these steps to compile and run:   1. Download the zip file and extract 2. Change directory to “project” located inside “Investment App” folder 3. Compilation command: javac frontend/InvestmentApp.java 4. Execution command: java frontend.InvestmentApp   To be tested using the windows OS. |
|  |