Team Pythia

Algorithmic Stock Trading

Clement Cole Enrique Torres

Christopher Roche Elijah Adedapo

Final Design Document

Version 1.0

October 27, 2016

Contents

I. Requirements

A. Functional Requirements

1. The system will have a User Interface for hardware-based unit to select which stock's forecasted lows and high to be displayed.
    a. Elijah Adedapo
2. The system will have a User Interface for the software Based computer unit to select which stock's forecasted lows and high to be displayed.
    a. Elijah Adedapo
3. The system will have a data source for the hardware-based unit from external web scraping tool to acquire stock data.
    a. Christopher Roche
4. The system will have a data source for the software-based unit from external web scraping tool to acquire stock data.
    a. Christopher Roche
5. The system will have a data source formatter for input into the hardware-based unit.
    a. Christopher Roche
6. The system will have a data source formatter for input into the software-based unit.
    a. Christopher Roche
7. The system will have a data source formatter for output from hardware-based unit into the computer system server.
    a. Clement Cole
8. The system will have a data source formatter for output from software-based unit into the computer system server.
    a. Enrique Torres
9. User will be able to control the variety of stock output data to be display in User Interface for hardware-based unit.
    a. Elijah Adedapo
10. User will be able to control the variety of stock output data to be display in User Interface for software-based unit.
    a. Elijah Adedapo
11. Hardware-based unit output will be displayed via a User Interface.
    a. Elijah Adedapo
12. Software-based unit output will be displayed via a User Interface.
    a. Elijah Adedapo
13. Hardware-based unit must predict the following daily highs and lows of the provided stocks.
    a. Clement Cole
14. Software-based unit must predict the following daily highs and lows of the provided stocks.
    a. Enrique Torres

B. Non-Functional Requirements

1. The system shall have a User Interface Display to show stock prediction report via graphic user interface for hardware-based unit.
    a. Elijah Adedapo
2. The system shall have a User Interface Display to show stock prediction report via graphic user interface for software-based system.
    a. Elijah Adedapo
3. The system must be able to display the performance of both hardware and software implementations of the algorithm in terms of speed and latency for both the hardware-based and software-based units.
    a. Elijah Adedapo
4. Hardware-based unit must be available during pre-trading hours and regular trading hours.
    a. Clement Cole
5. Software-based architecture must be available during pre-trading hours and regular trading hours.
    a. Enrique Torres
6. The entire system shall be small enough to fit provided enclosure in the stock trading ground.
    a. Enrique Torres

C. Constraints

1. Network
    a. Constraint: The product will use the current internet provider service for the University of North Texas to retrieve data from the internet using its respective internet service provider.
    b. Justification: This premise is due to the easy access to several mediums (wired/Wi-Fi) University's network via student user name and password.
    c. Fit Criterion: The product shall be compatible to any medium for internet access (Wi-Fi / wireless).
2. Operating System
    a. Constraint: The group will use both Windows and Linux based operating systems
    b. Justification: Both hardware and software system development aspects can be done on Linux or Windows architectures.
    c. Fit Criterion: Most users/customers use analytic tools that are compatible with Linux and Windows based architectures. The product will be fully compatible with both operating systems.
3. Stock Data
    a. Constraint: Yahoo Finance API written in python programming language will be used to scrape the data parameters or terminals from the internet.
    b. Justification: This is because the Yahoo Finance API is the most reliable and tested API by the group and confirmed to work efficiently.
    c. Fit Criterion: A wrapper class written in C++ or any other system based programming language can be used for the Yahoo Finance API written in python.

4. Version Control
    a. Constraint: The group will utilize GitHub for version control.
    b. Justification: Github is standard and free.
    c. Fit Criterion: The group can utilize the private feature on the web application which is free for student. This will allow the group to work on the project remotely while still contributing significantly to the entire project.

II. Specifications

A. Functional Requirement Specifications

1. The user interface will be created using Xamarin studio IDE .Net application platform to create a stock prediction report. The interface will be mainly programmed in C# because its libraries are compatible across multiple platforms such as Android, IOS, and Windows operating systems. The system will be able to receive data from hardware implementation of the analysis algorithm and display its results.

2. The user interface will be created using Xamarin studio IDE .Net application platform to create a stock prediction report. The interface will be mainly programmed in C# because its libraries are compatible across multiple platforms such as Android, IOS, and Windows operating systems. The system will be able to receive data from software implementation of the analysis algorithm and display its results.

3. Historical stock data will be pulled from Yahoo Finance via the Yahoo Finance API implementation for Python. Data will be gathered using the stockName.get_historical("start", "end") function. The end date will be the date previous to the day the script is being executed. The start day will be five years previous to the date the script is being executed. This can be done using the Datetime Python library.

4. Historical stock data will be pulled from Yahoo Finance via the Yahoo Finance API implementation for Python. Data will be gathered using the stockName.get_historical("start", "end") function. The end date will be the date previous to the day the script is being executed. The start day will be five years previous to the date the script is being executed. This can be done using the Datetime Python library.

5. The historical data will be loaded into a data frame using the Pandas Python library for parsing. The columns of data needed for analysis by the algorithm will be saved into a comma separated values or text file that can be passed to the FPGA for hardware algorithm implementation.

6. The historical data will be loaded into a data frame using the Pandas Python library for parsing. The columns of data needed for analysis by the algorithm will be saved into a comma separated values or text file that can be passed to the software algorithm implementation.

7. The data resulting from the hardware-based analysis of the historical stock data will be passed back to the computer system from the FPGA via a data bus. This data will be saved into a comma separated values or text file where it can then be accessed by the User Interface system for display to the user.

8. The data resulting from the software-based analysis of the historical stock data will be formatted using C++. The data will then be saved into the comma separated values file where it can be accessed by the User Interface system for display to the user.

9. The user interface will have a new request functionality to update different stocks that the user wants to request. The new request functionally will only return new historical stock data that is not already in the database. The New request function will be implemented by the user interface platform created using Xamarin studio IDE .Net application platform for the hardware-based algorithm implementation.

10. The user interface will have a new request functionality to update different stocks that the user wants to request. The new request functionally will only return new historical stock data that is not already in the database. The New request function will be implemented by the user interface platform created using Xamarin studio IDE .Net application platform for the software-based algorithm implementation.

11. The expected results of the data that is implemented on the hardware-based algorithm will be fed into a database that will be accessed as a library in the Xamarin studio IDE .Net application platform. The database will feed automatically to the .Net application to eliminate the need of having to re-import the file each time the database is updated.

12. The expected results of the data that is implemented on the software-based algorithm will be fed into a database that will be accessed as a library in the Xamarin studio IDE .Net application platform. The database will feed automatically to the .Net application to eliminate the need of having to re-import the file each time the database is updated.

13. The hardware-based implementation of the prediction algorithm will consist of a genetic algorithm running on a field-programmable gate array (FPGA). The FPGA to be used will be in the Virtex-5 or 6 family. The algorithm will take five years' worth of historical data on a given stock and predict the high and low sale values for the stock for the following trading day. The genetic algorithm involves the creation of parent and children pairs that are crossed over via a mathematical function. The fitness of each child is determined through a mathematical function and the children with the highest fitness are kept. This is repeated until the algorithm can determine a percent change from one day to the next based on historical context. The percent change can be used to estimate the high and low values for the following trading day. This analysis will be performed on ten stocks at a time but the user may change the ten stocks between each running of the algorithm. (See: "Stock price prediction using genetic algorithms and evolution strategies" by Ganesh Bonde and Rasheed Khaled)

14. The software-based implementation of the prediction algorithm will consist of a genetic algorithm programmed in C++. The algorithm will take five years' worth of historical data on a stock and predict the high and low sale values for the stock for the following day. The genetic algorithm involves the creation of parent and children pairs that are crossed over via a mathematical function. The fitness of each child is determined through a mathematical function and the children with the highest fitness are kept. This is repeated until the algorithm can determine a percent change from one day to the next based on historical context. The percent change can be used to estimate the high and low values for the following trading day. This analysis will be performed on ten stocks at a time but the user may change the ten stocks between each running of the algorithm. (See: "Stock price prediction using genetic algorithms and evolution strategies" by Ganesh Bonde and Rasheed Khaled)

B. Non-Functional Requirement Specifications

1. Some graphical features may be implemented in the user interface of the software-based system to make viewing statistical data easier to view for the user such as trend lines, and distribution curves. The user may also want to look at the trend and prediction values of previous calculations in a graph. Minimum information that will be displayed is the previous day closing value for each stock and the projected high and low sale value for each stock on the following trading day. This will allow the trader to make an informed decision on their stock positions.

2. Some graphical features may be implemented in the user interface design of the hardware-based system to make viewing statistical data easier to view for the user such as trend lines, and distribution curves. The user may also want to look at the trend and prediction values of previous calculations in a graph. Minimum information that will be displayed is the previous day closing value for each stock and the projected high and low sale value for each stock on the following trading day. This will allow the trader to make an informed decision on their stock positions.

3. The user interface will be able to display the results of the performance of both hardware and software based systems to determine their latency at performing the algorithm. This will allow the end user to make an informed decision as to which system performs the best; hardware or software based. This can be done by computing the clock speed at which the FPGA completes computation of the genetic algorithm. For the software-based implementation, a timing program will be generated on top of the algorithm to compute the speed of the genetic algorithm.

4. Testing of the hardware-based algorithm should eliminate all possible bugs. Running the algorithm should result in analysis that is consistently accurate. Adequate power supply and cooling should ensure the FPGA does not overheat and is constantly available to the stock trader. Outages on Yahoo Finance's servers pose the only reliability factor the engineers have

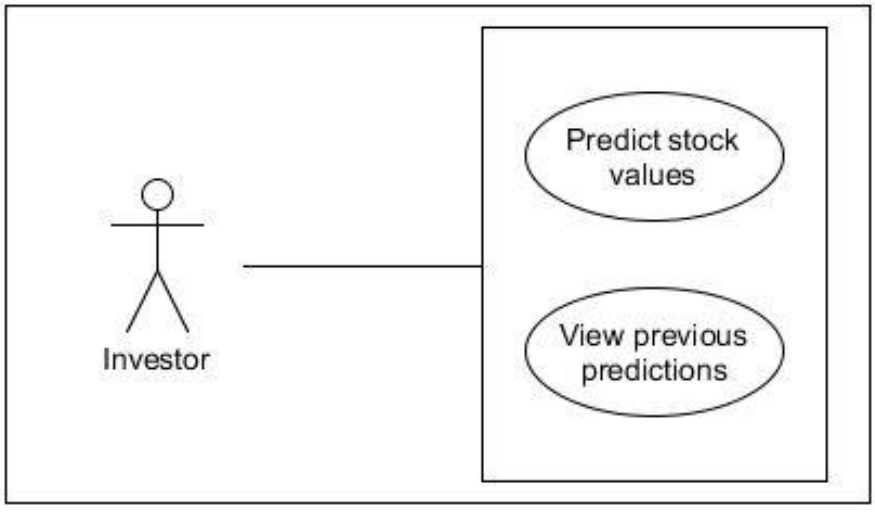
no control over. In this event the system will give the trader the option of analyzing the most recent data available locally. Trading hours are from 8:30AM to 3:00PM CT. The system should be available during pre-trading hours as well as during the trading day to update predictions.

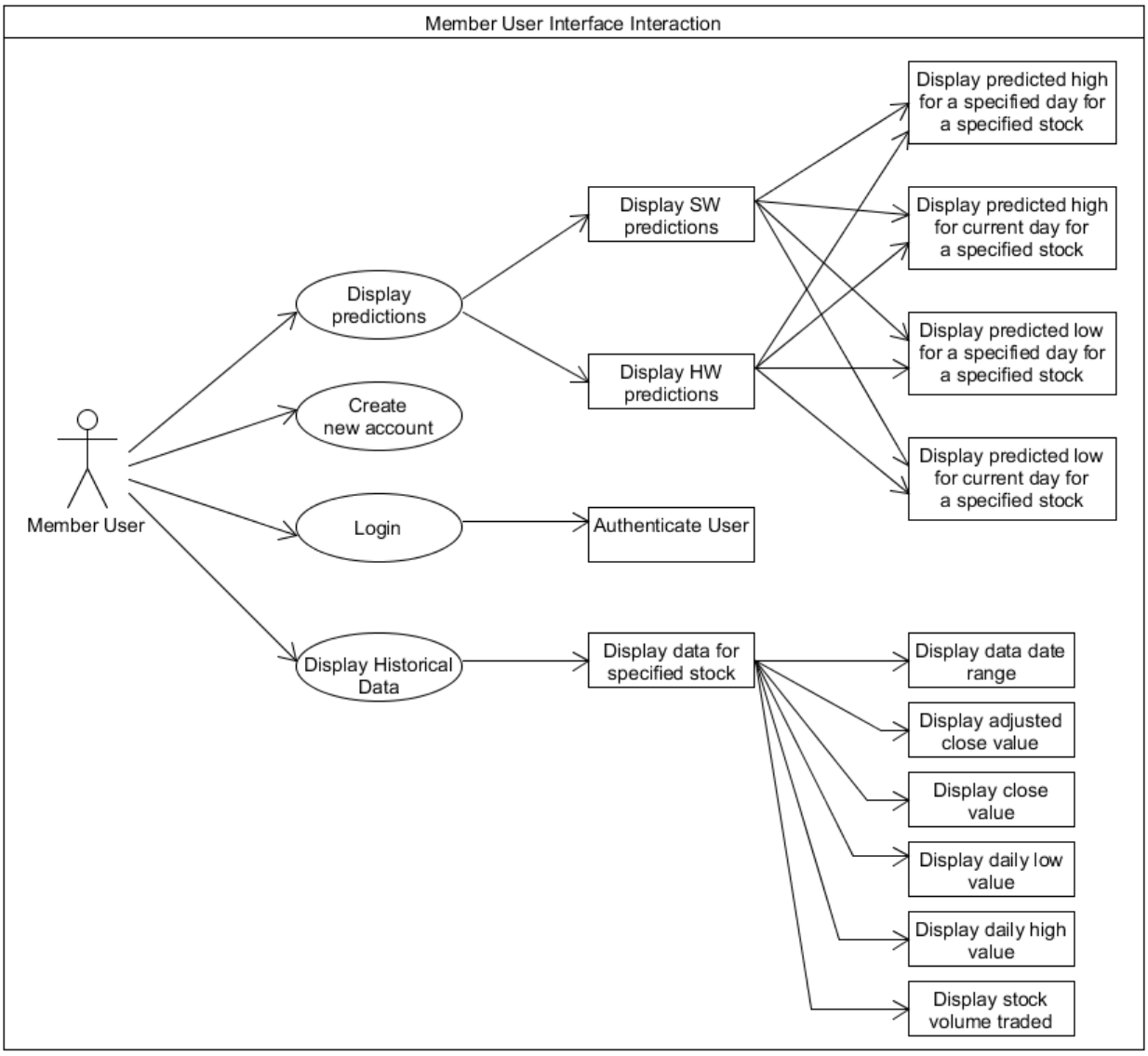
5. Testing of the software-based algorithm should eliminate all possible bugs. Running the algorithm should result in analysis that is consistently accurate. Adequate power supply and cooling should ensure the FPGA does not overheat and is constantly available to the stock trader. Outages on Yahoo Finance's servers pose the only reliability factor the engineers have no control over. In this event the system will give the trader the option of analyzing the most recent data available locally. Trading hours are from 8:30AM to 3:00PM CT. The system should be available during pre-trading hours as well as during the trading day to update predictions.

6. To maximize the efficiency of the prediction algorithm, the system needs to be as close to the stock trading floor as possible. This will reduce latency between the system and the exchange during buy and sell orders. The field-programmable gate array to be used for hardware analysis will be in the Virtex-5 or 6 family. The full system should not be much larger than the average commercially available computer tower. The full system needs to be able to fit on the trading floor under a desk. Cat-6 Ethernet cable will be sufficient to process information from the stock market floor.

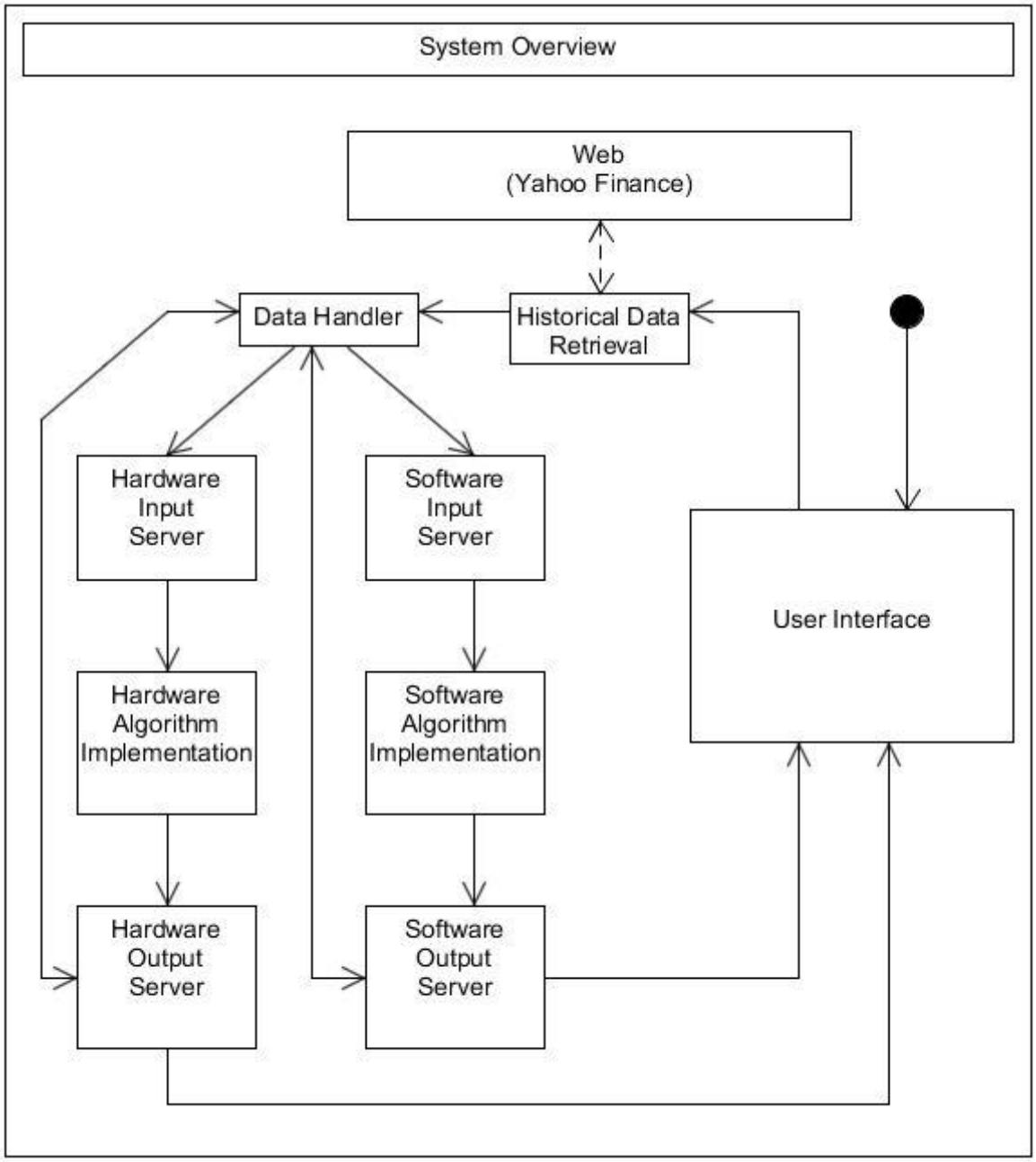
III. Project Design

A. Use Cases

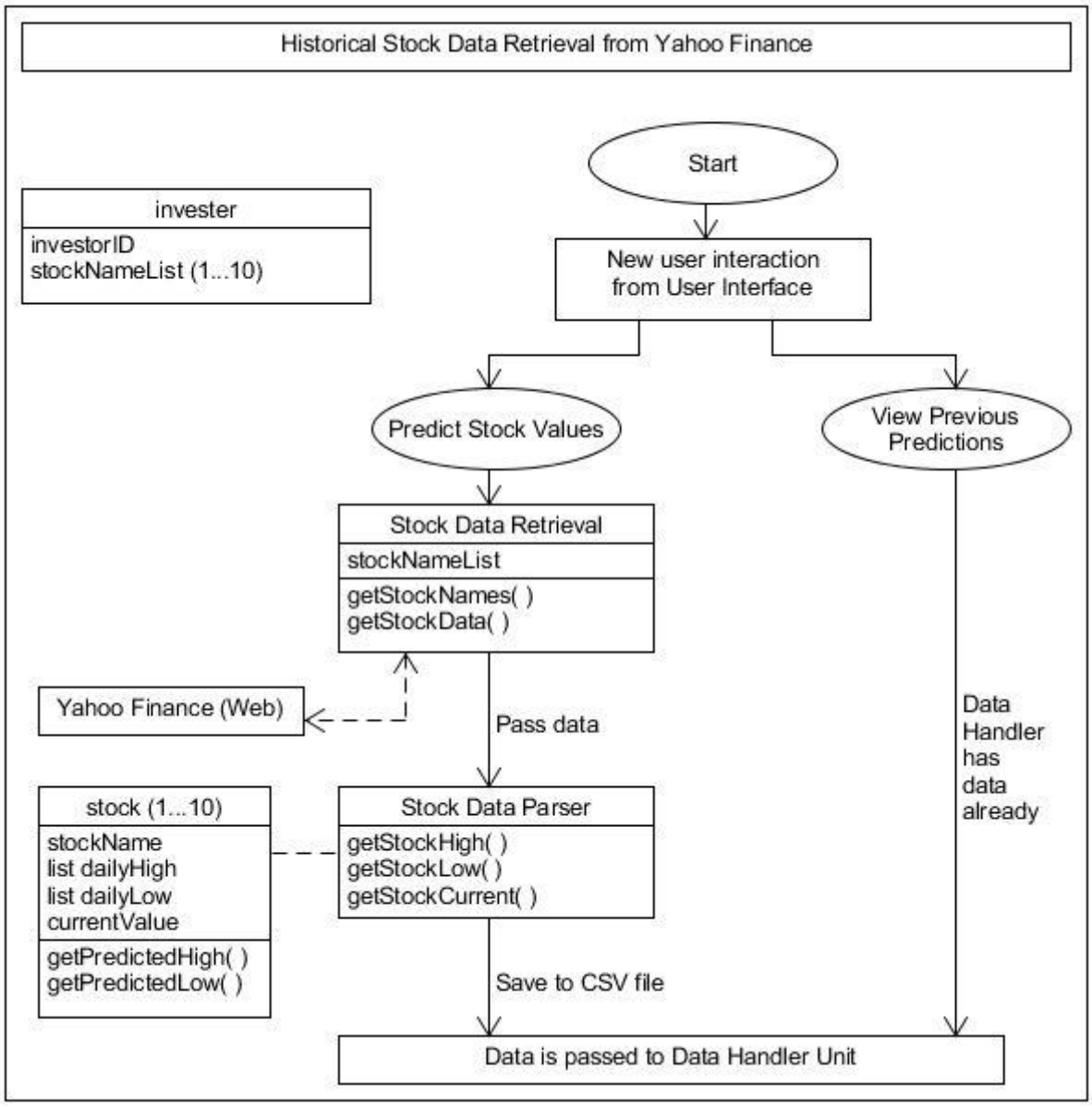


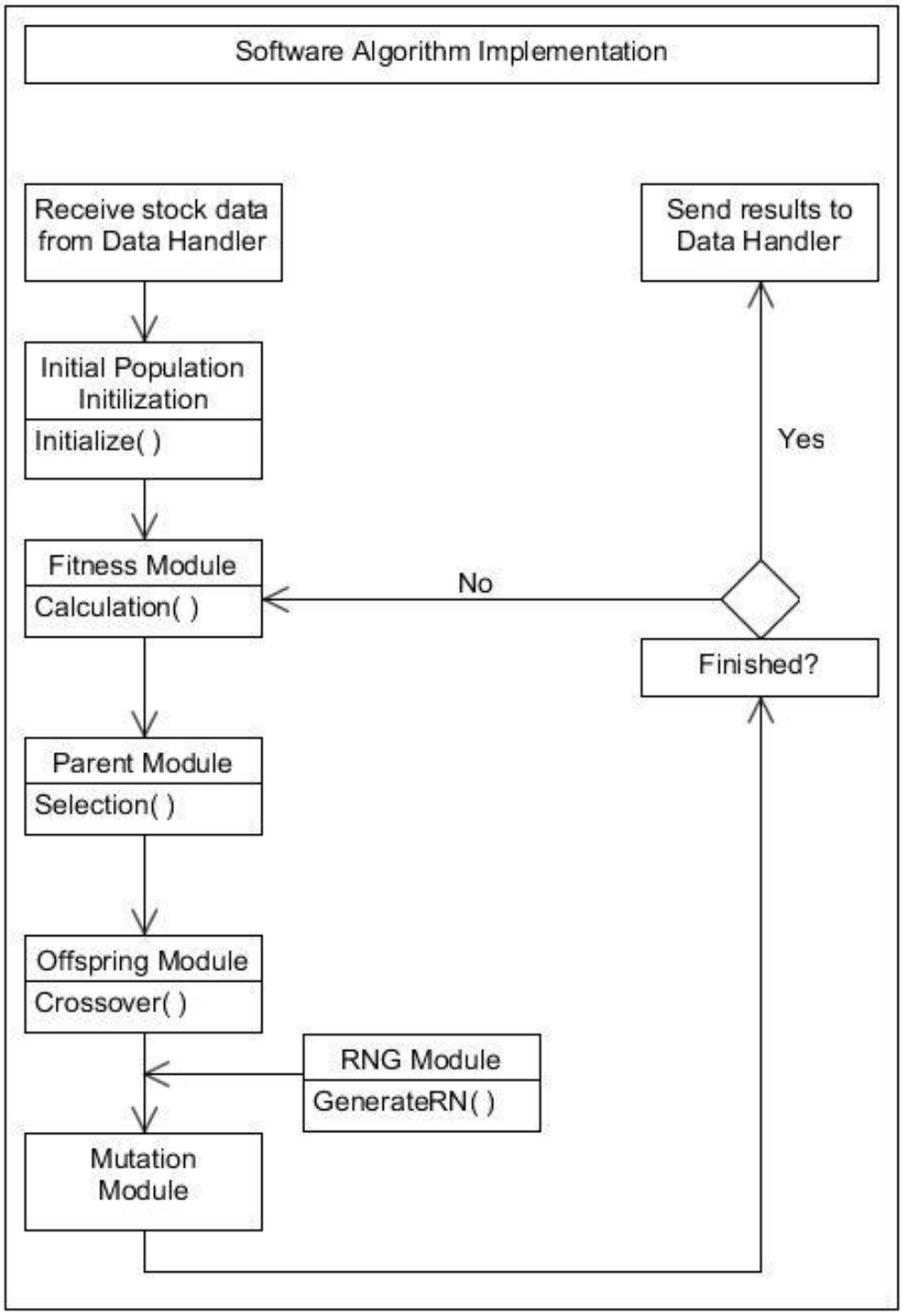

Member User Interface Interaction
Member User
Display predictions
Create new account
Login
Display Historical Data
Display SW predictions
Display HW predictions
Authenticate User
Display data for specified stock
Display predicted high for a specified day for a specified stock
Display predicted high for current day for a specified stock
Display predicted low for a specified day for a specified stock
Display predicted low for current day for a specified stock
Display data date range
Display adjusted close value
Display close value
Display daily low value
Display daily high value
Display stock volume traded

B. System Overview

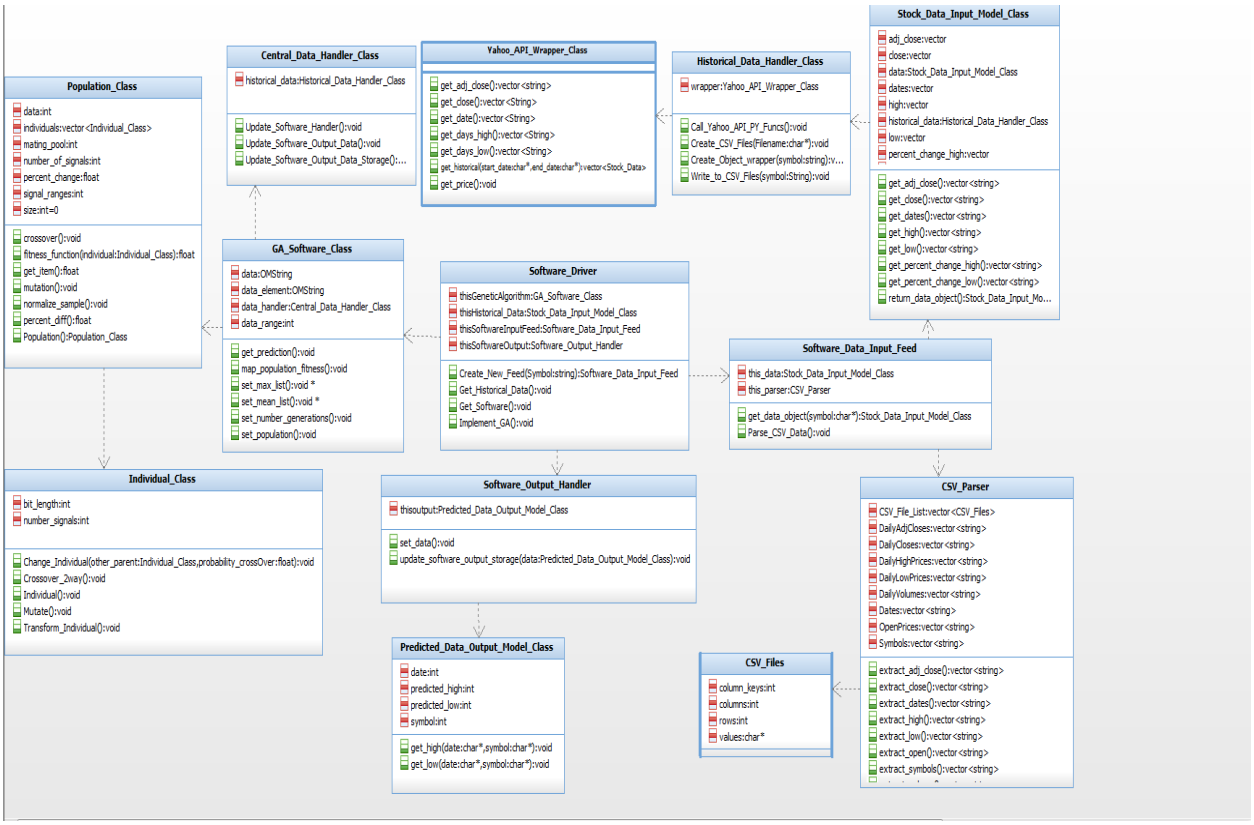


C. Class and State Diagrams

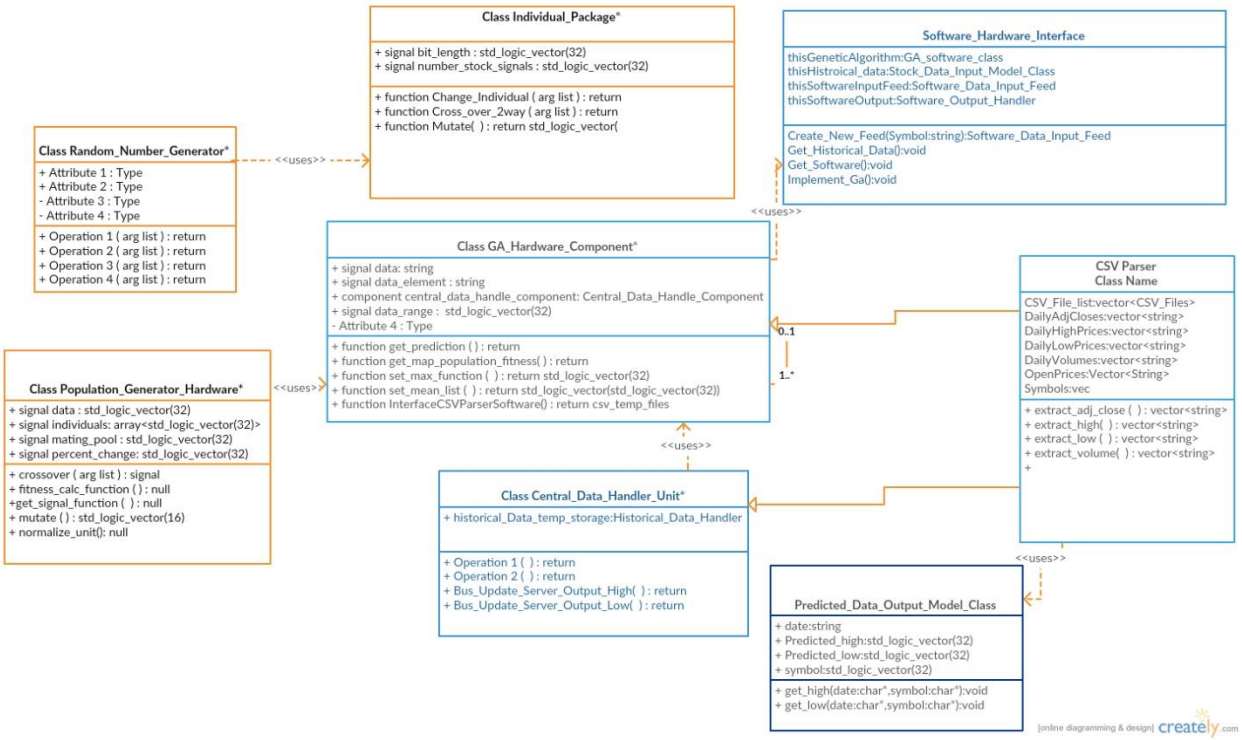



Software Algorithm Implementation
Receive stock data from Data Handler
Initial Population Initilization
Initialize( )
Fitness Module
Calculation( )
Parent Module
Selection( )
Offspring Module
Crossover( )
RNG Module
GenerateRN( )
Mutation Module
Finished?
No
Yes
Send results to Data Handler

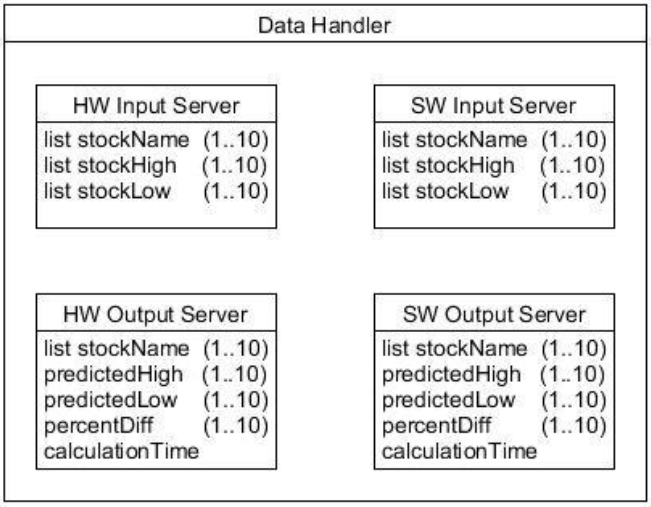
Software Class Diagram

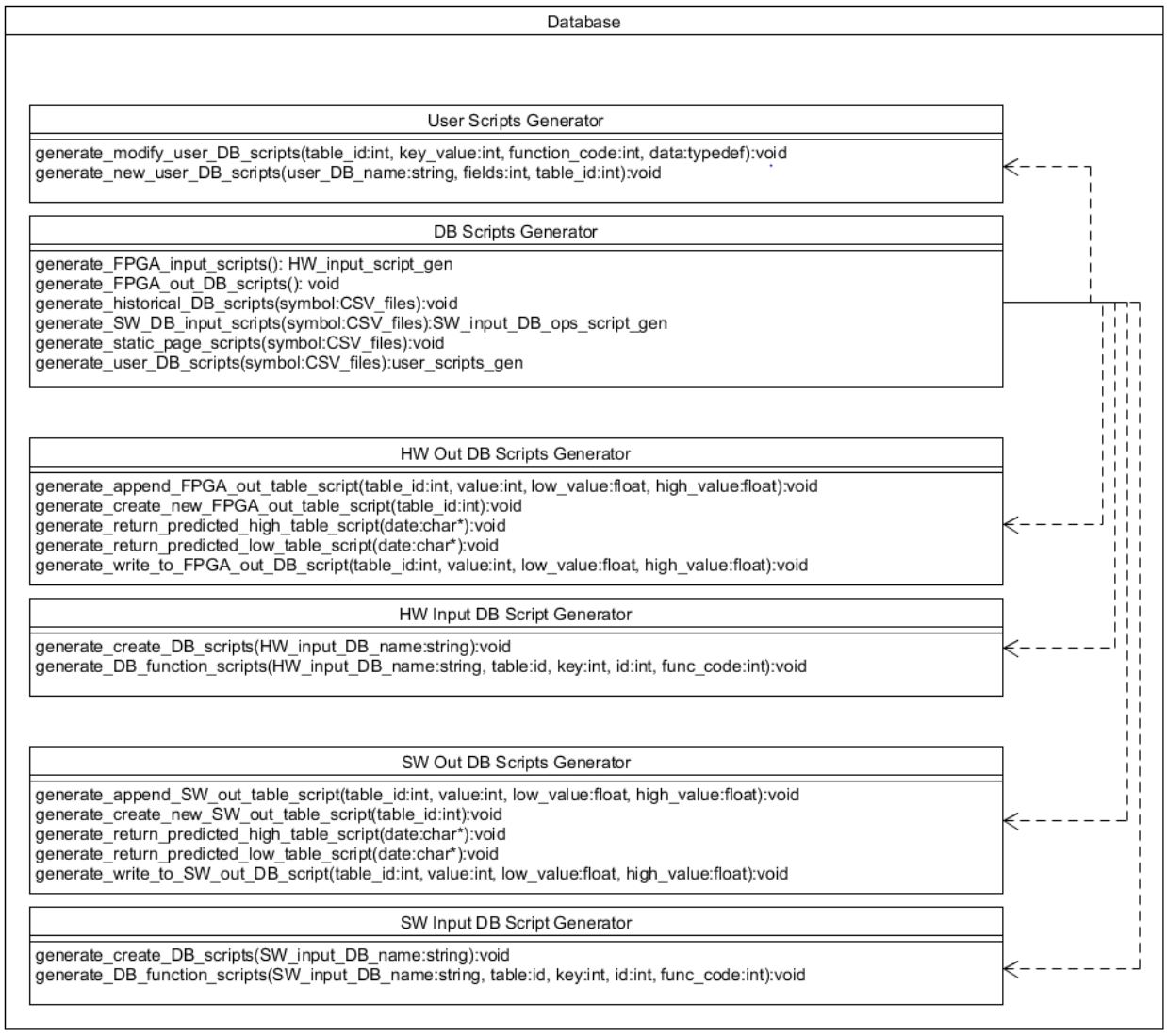


Hardware Class Diagram

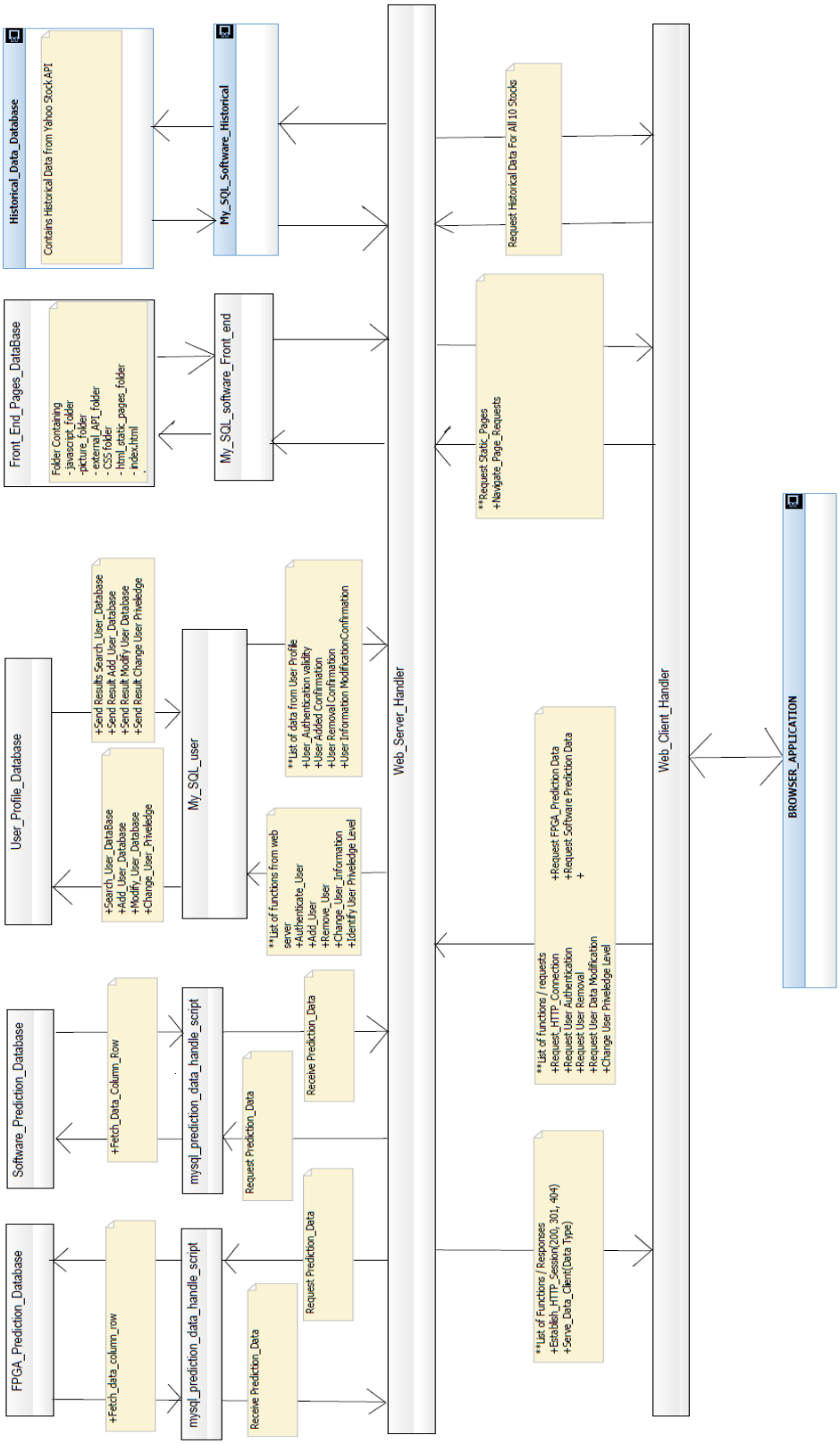


Data Handler and Servers

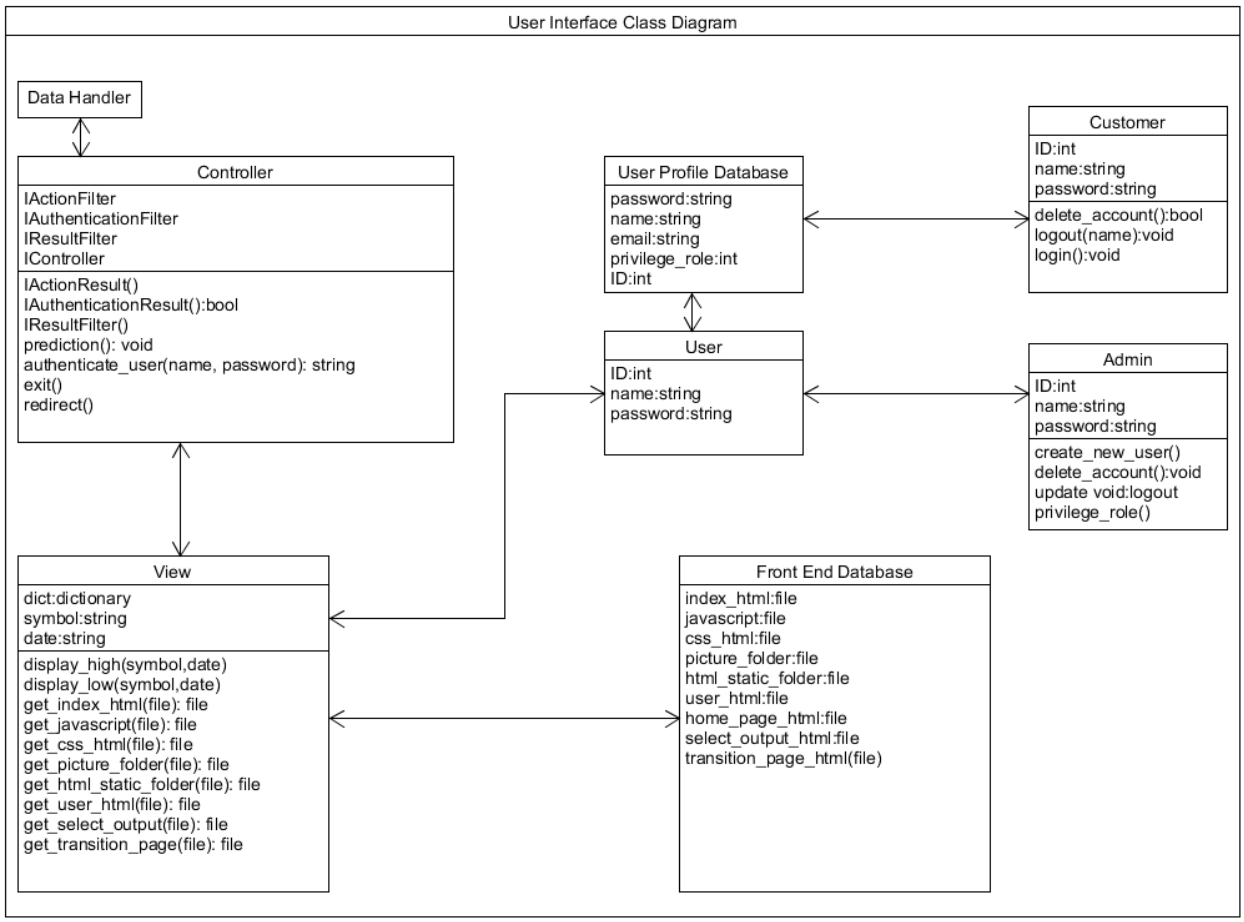


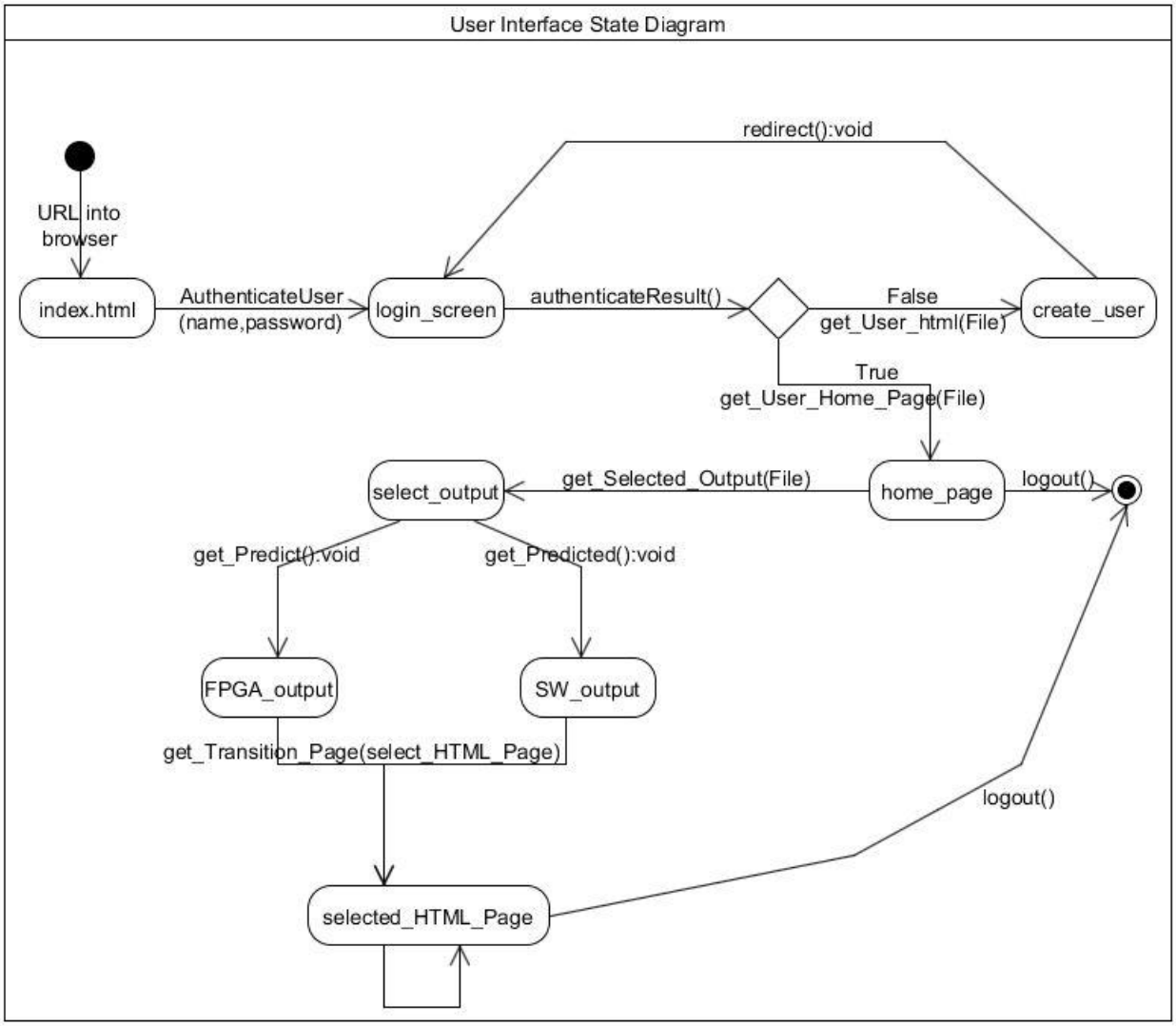


Server Component Design

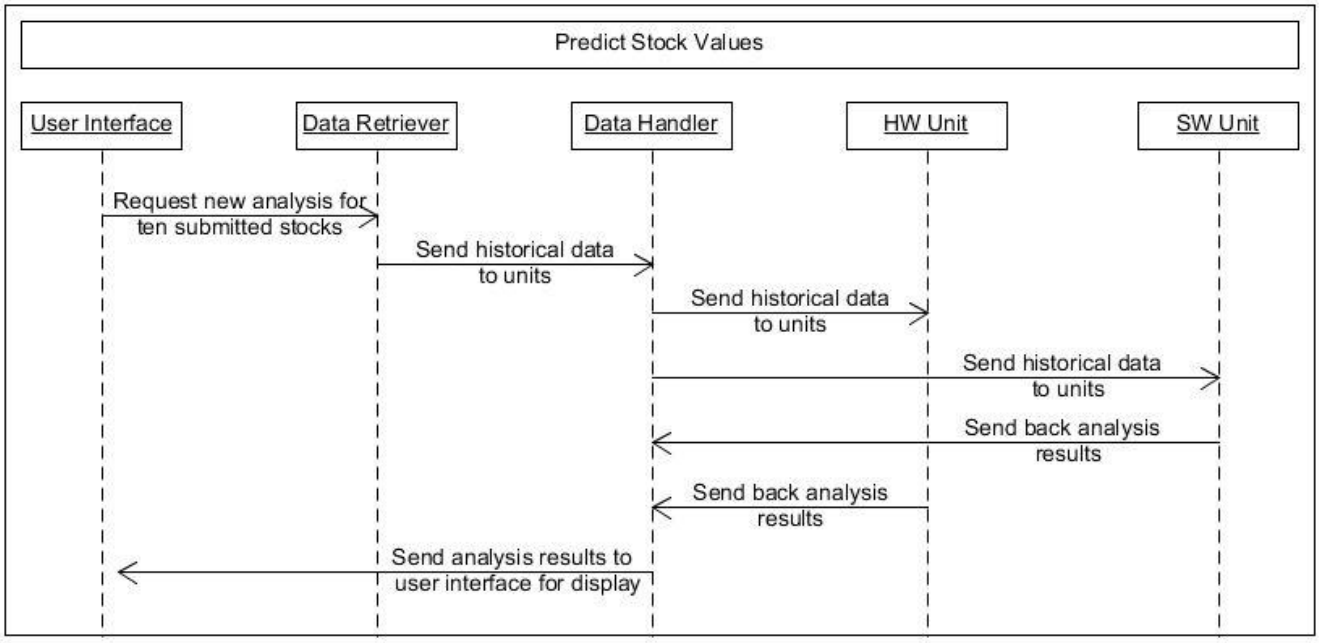


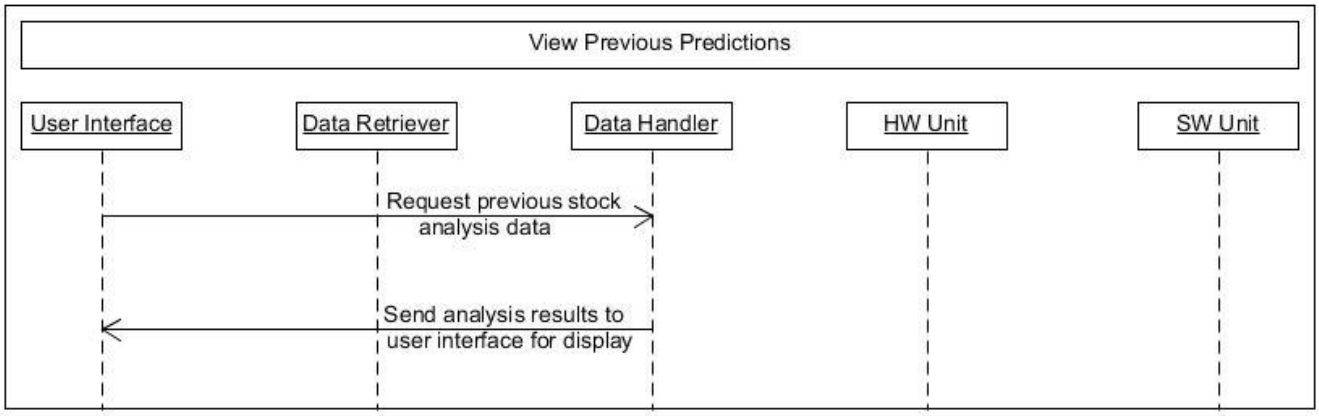
User Interface



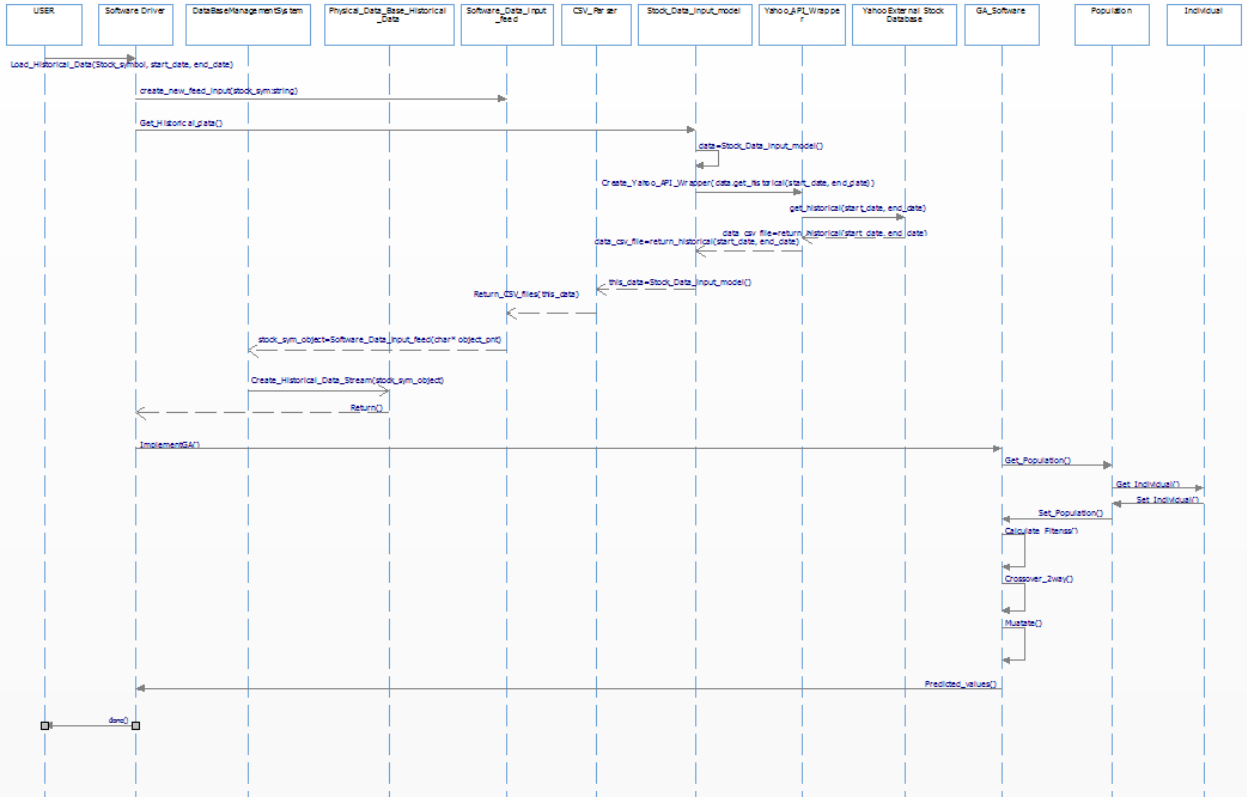

User Interface State Diagram
redirect():void
URL into browser
index.html
AuthenticateUser (name,password)
login_screen
authenticateResult()
False
get_User_html(File)
create_user
True
get_User_Home_Page(File)
select_output
get_Selected_Output(File)
home_page
logout()
get_Predict():void
get_Predicted():void
FPGA_output
SW_output
get_Transition_Page(select_HTML_Page)
logout()
selected_HTML_Page

D. Sequence Diagrams

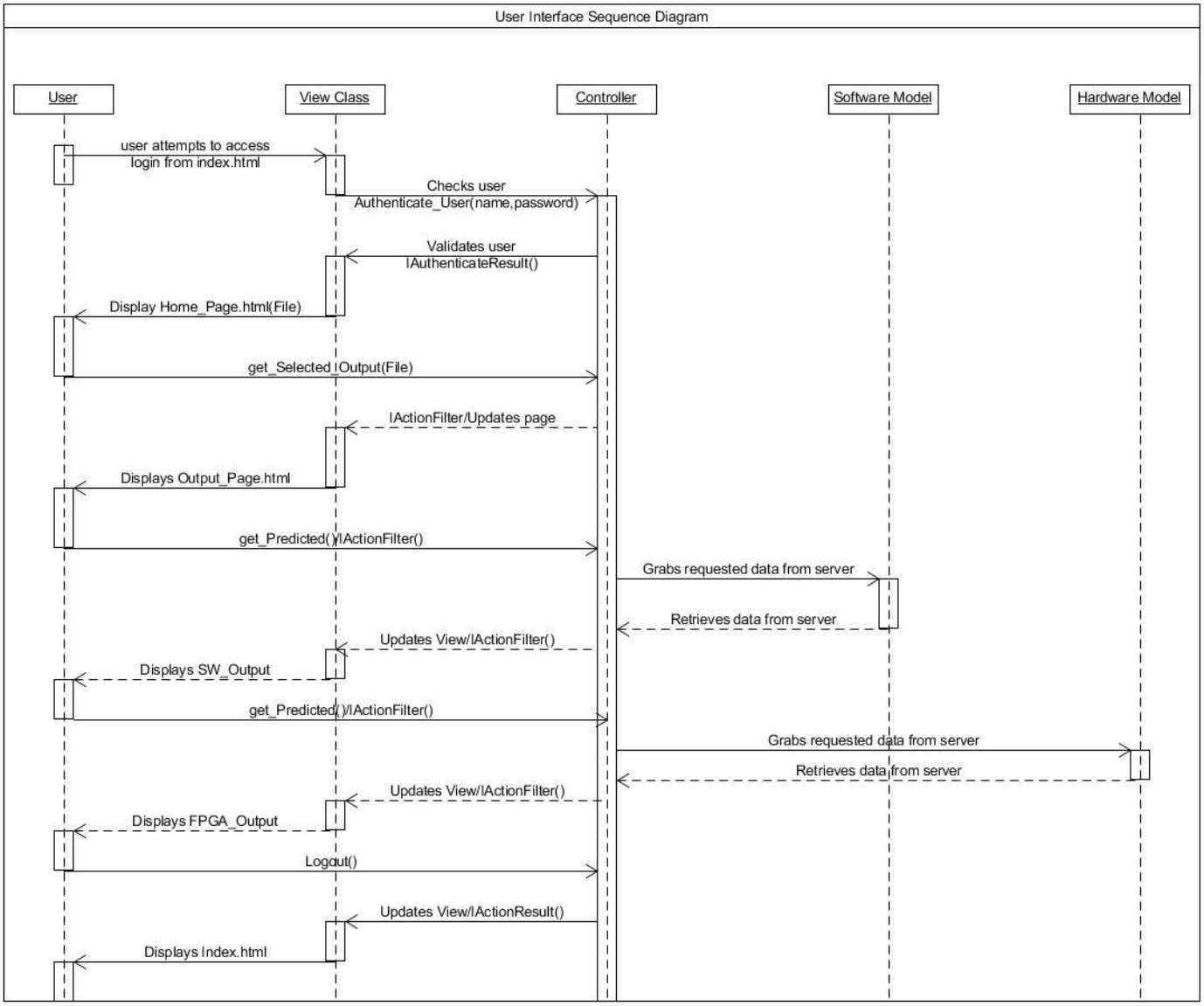




Software Component



Hardware Component



IV. Citation

Bonde, Ganesh, and Rasheed Khaled. "Stock price prediction using genetic algorithms and evolution strategies." *Proceedings of the International Conference on Genetic and Evolutionary Methods (GEM).* The Steering Committee of the World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2012