UNIVERSITY OF WATERLOO
DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

Detailed Design
StealthWealth
(Stock Price Predicting App)

Group 2020.56

Prepared by

Youjing Li
ID 20602178
userid y864li

Iruj Fatima
ID 20608774
userid i2fatima

Shumeng Zhang
ID 20621538
userid sazhang

Seyeon Kim
ID 20614712
userid s357kim

Consultant: Alex Wong
25 June 2019

Table of Contents

1 High-level Description

1.1 Motivation

Stock trading plays a vital role in economic development. For starters who wish to enter the stock market, it is important to educate themselves of potential risks and trading strategies prior to stepping foot in the investing world [1]. Many similar applications are available with distinct purposes, whether it is informative, predictive, or educational. However, our platform aims to achieve all three goals. It equips ambitious individuals with enough financial background to start investing their money wisely.

1.2 Project Objective

The objective of this project is to design a stock simulation application which provides an all-in-one educational platform that features stock information, recommendation, and prediction.

1.3 Block Diagram

The proposed solution is a system that generates stock predictions considering user-centered design methods, optimization strategies, machine learning concepts, and cloud technologies. The system architecture consists of four main subsystems: data collection, data storage, data prediction, and data visualization. A simple data flow diagram is illustrated to show the data flow in between the subsystems in Figure 1 and the individual components are listed in Figure 2.

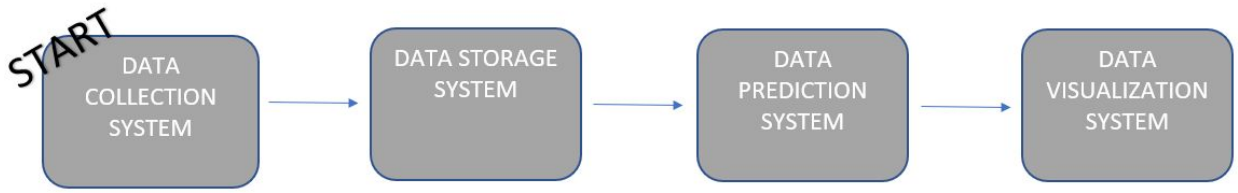

Figure 2. Block Diagram Outlining Subsystem Interactions

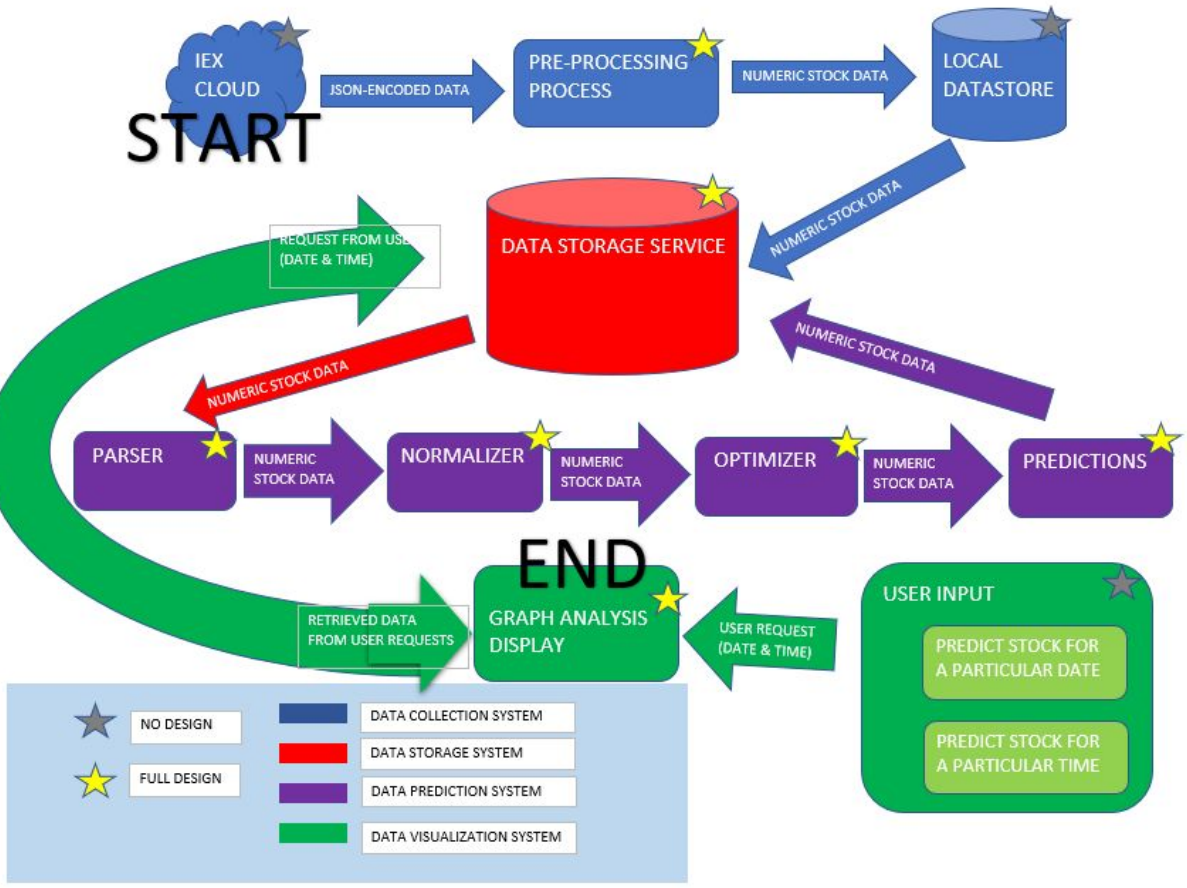

Figure 2. Block Diagram Outlining Subsystem Interactions

Table 1. Provides a detailed description of the components shown in Figure 2

Component	To be designed	Description
IEX Cloud	No	IEX Cloud provides financial data and its API is REST based with resource-oriented URLs.
Pre-processing Process	Yes	Upon receiving JSON-encoded stock information, the pre-processing script uses python modules iexfinance and pandas to extract and clean stock data.
Local Datastore	No	The clean data is cached locally in CSV file format.
Parser	Yes	The stored data is parsed into Python for analytical operations.

Normalizer	Yes	A scaler is defined for normalization; training and test data are also defined and reshaped.
Optimizer	Yes	Recurrent Neural Networks are designed to increase prediction accuracy and processing speed.
Predictions	Yes	Future stock trends are extracted and transported back to the database.
Graph Analysis Display	Yes	Visualizations are updated upon user requests and are based off data collected from Data Collection System and Trend Prediction System.
User Input	No	User inputs are collected to determine the time range to graph for the visualization output.

2 Requirements

2.1 Functional Requirements

Table 2. A list of all of the functional requirements describing the operations of the application

Subsystem	**Specification**	**Essential**	**Description**
Data Storage	Storage service	Yes	The storage mechanism must have the ability to store up to 30 years of historical data for easy access, manipulation, and modification.
Data Collection	Stock market data	Yes	The data collection system must be able to extract and clean raw stock data from all major stock indices. This data includes each stock's opening, closing, highest, and lowest price of the day.

	Accuracy	Yes	The collected data must be true to the real-time changes in stock price. It must be collected at a rate that provides sufficient information to determine the opening, closing, highest, and lowest price of each stock, while meeting all timing constraints.
Data Visualization	Predictions	Yes	The application should display future predictions of approximately three months of the stock market. These predictions include but are not limited to: individual price, trends, and behaviour relative to other stocks.
	Graphical Representation	Yes	Must be able to display stock market data trends.
Data Prediction	Speed	Yes	CPU utilization differs based on several factors. For instance, the hour of the day and the different processes that are being run. These factors can excessively reduce performance of the code. In order to limit the compile times and access times, additional CPUs (or GPUs) need to be able to share the workload.
Machine Learning	Algorithm	Yes	A suitable machine learning algorithm must be chosen to implement the time-series forecast model. It must be able to make accurate long term predictions for a large set of data. The algorithm should take into consideration past values and past forecasting errors to predict future values.

	LSTM Dataset	Yes	Machine learning algorithm is dependent on large, diverse datasets. Without its presence, the algorithm can not form relationships and clusters that are required for data exploration, data mining, and machine learning.

2.2 Non-functional Requirements

Table 3. A list of all of the non-functional requirements describing the general characteristics

Subsystem	**Specification**	**Essential**	**Description**
Data storage	Security	Yes	Prevent unauthorized access of the data storage to securely store data. Also, prevent capacity or overloading issues that could lead to corrupting the database.
	Manageability	No	Data must be stored and organized using a technique that is simple to read and write files to. The data storage technique must be relatively easy to interpret as well.
	Data Integrity	Yes	Data is protected from corruption of invalid data. The data remains accurate through daily polling.
Data Collection	Speed	No	Data must be collected and stored at a relatively high rate so that real-time updates can be provided in a timely manner.

	Historical Data	Yes	Must be able to collect stock market data from the past 30 years in order to make accurate future predictions.
Data Visualization	Customize Dashboard	No	User is able to customize their dashboard with different stock indices. Some of these indices include but are not limited to: Nasdaq and TSX.
Data Prediction	Prediction Period	Yes	Must be able to predict at least three months of stock price trends to a high degree of accuracy.

3 Detailed Design

3.1 Software Subsystem Design

The software subsystem includes data extracting system, data storage system, data processing system, and data visualization system. In this section, the choice of programming language and software libraries that will be used for the project will be explained.

3.1.1 Programming Language

When selecting a programming language for the project, several criteria need to be considered. The main criterion is to use a language that is the most effective for machine learning since the project incorporates machine learning concepts. Another thing to consider is that the project will be dealing with a large amount of stock data. After taking these criteria into consideration, Python, C++, and Java are found to be the most effective languages for machine learning processing, where Python is the most suitable language for this project.

About 57% of machine learning developers use Python [3]. Java can be used for machine learning, but it is more prioritized for working with network security. C++ is another language that can be used, yet it is mostly used for making VR/AR applications. Python supports object-oriented programming, procedure and functional oriented styles of programming [4]. Python is simple and has an extensive collection of libraries such as PyTorch, TensorFlow, and Pybrain. As a result, it offers a faster way to build high performing algorithms, and it will be ideal for this project.

3.1.2 Software Libraries

In order to compute the machine learning process, machine learning libraries should be used. There are closed source tools such as Microsoft Azure or Amazon Machine Learning, but open source libraries are a great starting point [3]. Instead of building a stock prediction system from scratch which will take a substantial amount of time and effort, software libraries can help to build the dataset, and provide various functions. Python provides a lot of software libraries for machine learning. For this project, the three main libraries which are TensorFlow, Pandas, and Numpy will be utilized.

TensorFlow which was developed by Google Brain is an open source library for processing large-scale machine learning [5]. One of the advantages of using it is an abstraction. If an algorithm had to be implemented from scratch, it will take days to complete it. However, TensorFlow provides functions for building recurrent neural networks and computing data. It also has excellent debugging features, thus if there is a bug in the program, it will be easy to find. One important thing to keep in mind is that TensorFlow only supports Nvidia GPUs [5].

Pandas is another library that will be frequently used. It is a powerful tool for creating flexible data structures [6]. For this project, to make data easy to manipulate and analyze, the data structure needs to be created beforehand.

Last but not least, Numpy is a Python package for scientific computing [7]. Numpy will be used to compute mathematical operations on large data sets including arrays, matrices and so on.

3.2 Data Retrieval

To analyze the stock market, stock data should be retrieved from the internet or any other external sources, and stored in a data storage service. There are several libraries that help to extract stock data. For this project, iexfinance, which is a Python SDK for IEXCloud, will be used to extract past and real-time stock data.

3.2.1 Data Access

Accessing the stock data using iexfinance is very simple. The stock data can be extracted daily using get_historical_data function or every minute using get_historical_intraday [8]. The stock data contains the volume, the company's name, as well as low, high, opening and closing prices. Figure 3 shows how Tesla's stock data from 2014 to the present is retrieved.

```
from datetime import datetime
from iexfinance.stocks import get_historical_data


start = datetime(2014, 1, 1)
end = datetime.now()

df = get_historical_data("TSLA", start, end)
```

Figure 3. Pseudocode for extracting Tesla stock data

After all the stock data has been extracted, they will be stored in a data storage service which will be explained in later sections. The stored data will be accessed from storage when computing machine learning processes to predict future stock value or when visualizing them for users.

3.2.2 Real-Time Data

The iexfinance SDK also has a function to get the current stock data. Using this functionality, real-time stock data can be used to predict future stock value or give the users real-time stock chart. The benefit of using real-time data is that it minimizes risks when predicting future outcomes. To get the stock data continuously, the threading library can be used to call a function every x seconds. Figure 4 displays an implementation of polling Tesla stock data every five seconds.

```
from datetime import datetime
from iexfinance.stocks import get_historical_data
import threading

def realTimeStock():
    threading.Timer(5.0, realTimeStock).start()
    df = get_historical_data("TSLA", datetime.now())
    print(df)
```

Figure 4. Pseudocode for extracting Tesla stock data every five seconds

3.3 Cloud Service Provider

A stock market prediction application requires the storage and retrieval of thousands of data points. In order to store all of the data points gathered from various companies, an effective data storage mechanism is required. To achieve this, three public cloud service providers were considered: Amazon Web Services, Microsoft Azure, and Google Cloud. When comparing cloud service providers, many components can be analyzed. Some components analyzed in this section are data storage, computing services, and management tools.

3.3.1 Data Storage

All three service providers are capable of storing large volumes of data which is a requirement for this application. However, Google Cloud does not provide backup services and this could be

detrimental if a failure occurs. Since the occurrence of a failure can be considered rare, we will continue to compare all three options to determine the optimal one.

Before considering the pricing of data storage, we must take into consideration any provided free trials. These free trials will enable us to minimize costs when choosing the final service provider. Amazon Web Services provides users with a 12-month free tier for Amazon S3 which consists of 5GB of standard storage [11]. Microsoft Azure provides users with $260 in credits that can be used over the course of 30 days [12]. After the 30 days, users have access to their data storage for another 11 months. Finally, Google Cloud provides users with a 12-month free period in which they have $300 in credits to use [13]. Making use of these free trials can assist in reaching a final decision for an appropriate service provider.

Now, we consider the costs of storing additional data beyond the free tiers. Amazon S3 provides users with payment options for both frequently and infrequently accessed data. For our application, most of the data points will be frequently accessed depending on which companies are more commonly accessed. However, some companies or older data points may be infrequently accessed. For the analysis, we will consider both frequent and infrequent data access prices. Amazon S3 Standard Storage has a pay-as-you-go build. Table 1 below summarizes the payment structure for Amazon S3. All prices stated in the tables below are in Canadian dollars. Microsoft Azure Blob Storage is used to store large amounts of data and it has a similar pricing technique to Amazon S3. Table 2 summarizes the cost for using Azure Blob Storage. It consists of three different prices: hot storage, cool storage, and archive storage. Hot storage is used for frequently accessed data. Cool storage is used for infrequently accessed data, and archive storage is used for data that is rarely accessed but may still be required. Google Cloud Storage consists of fixed rates per GB used. Regional storage allows the storage of data in a specific region. For the stock market prediction application's purposes, regional storage is appropriate, and it is also the least costly. For data that needs to be infrequently and rarely accessed, nearline storage and coldline storage may be used respectively.

Table 4. Amazon S3 Standard Storage Cost [11]

Data Used Per Month (In TB)	**Cost Per Month (Per GB) (Frequent)**	**Cost Per Month (Per GB) (Infrequent)**

First 50	0.025	0.0138
Next 450	0.024	0.0138
Over 500	0.023	0.0138

Table 5. Microsoft Azure Blob Storage Cost [12]

Data Used Per Month (In TB)	**Cost Per Month (Per GB) (Hot Storage)**	**Cost Per Month (Per GB) (Cool Storage)**	**Cost Per Month (Per GB) (Archive Storage)**
First 50	0.0236	0.0128	0.0013
Next 450	0.0227	0.0128	0.0013
Over 500	0.0217	0.0128	0.0013

Table 6. Google Cloud Storage Cost [13]

Data Used Per Month (In TB)	**Cost Per Month (Per GB) (Regional Storage)**	**Cost Per Month (Per GB) (Nearline Storage)**	**Cost Per Month (Per GB) (Coldline Storage)**
N/A	0.023	0.013	0.007

From these tables, we can see that the range of prices between the three providers are relatively similar. No obvious outliers exist from the data presented. The first 50 terabytes of data using Amazon S3 is more costly than Microsoft Azure and Google Cloud, but as the number of used data increases, they begin to get closer in cost. Note however that Amazon S3 provides the most-costly option for infrequently accessed data.

3.3.2 Tools

All three of the cloud service providers contain compute services. AWS uses Elastic Compute Cloud (EC2). It contains large amounts of instances in which you can use Windows or Linux, and configure GPU instances, auto-scaling, and several other features. Microsoft Azure uses Virtual Machines and it supports both Windows and Linux. It provides relatively the same amount of functionality as EC2. However, it also allows users to have a free tier consisting of 750 hours per month for a total of 12 months. Lastly, Google Cloud uses Compute Engine and it

is a newer service. It does not contain as many services as Amazon or Microsoft, but it supports both Windows and Linux. In addition, the setup times for EC2, Virtual Machine, and Compute Engine are similar, and step-by-step tutorials can be found.

AWS provides server management tools. It allows users to control the visibility of different resources. AWS provides many other features that help enforce any policies that require restricting access. Likewise, Microsoft Azure contains management tools that help enforce policies. Google Cloud does not have server management capabilities. Also, all three service providers contain logging and monitoring capabilities. Another important comparison factor is the supported languages of the different cloud service providers' functions. The table below lists the supported languages. All three of the providers have Python capabilities and the stock market application will be implemented in Python.

Table 7. List of Compatible Languages [14] [15] [16]

Cloud Service Provider	**Supported Languages**
Amazon Web Services	C#, Go, Java, PowerShell, Python, Ruby
Microsoft Azure	C#, JavaScript, Java, PowerShell, Python
Google Cloud	JavaScript, Go, Python

3.3.3 Summary

After a thorough analysis of the three cloud service providers, a conclusion can be made on the most appropriate one for a stock market prediction application. Although Amazon Web Services is the most costly, the pay-as-you-go approach ensures that you are only paying for much you use. This payment approach applies to not only the data storage, but also the creation of instances. In addition, it provides a lot more services than Google Cloud, including automated backups of data. Amazon Web Services and Microsoft Azure are relatively similar in terms of functionality, however due to the large flexibility and team member familiarity, it is the cloud service provider we chose to go with.

3.4 Data Analysis

After saving the stock data into a data storage mechanism, the data should be extracted for predicting future stock values. Time series forecasting is one of the most significant areas of machine learning, and stock prices are one of the most interesting time series to predict. There are a number of machine learning techniques that can be used to predict future stock price. In this section, the machine learning technique for predicting stock prices will be described.

3.4.1 LSTM

The machine learning technique that will be implemented for the project is long short term memory (LSTM). LSTM is observed as the most effective solution for sequence prediction problems. LSTM is a kind of recurrent neural network (RNN) which is capable of learning long-term dependencies [9].

RNN is a network with a loop in it to persist information. The loop passes current information to the next step, allowing the step to use the past information to create an output. Figure 5 illustrates a diagram of RNN [9].

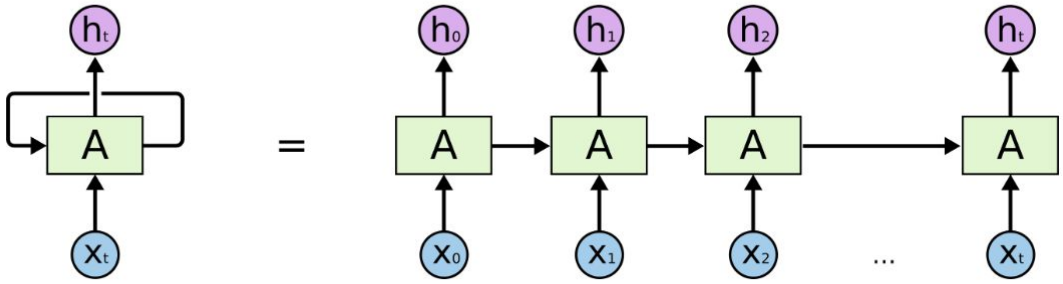

Figure 5. A diagram of the recurrent neural network [9]

In the diagram, A, x, and h represent a neural network, input, and output respectively. A neural network uses information from the previous step and current input to produce an output. The limitation that RNN has is that it cannot handle long-term dependencies, which means it cannot pick up information which was produced a long time ago. However, LSTM solves the problem.

LSTM, which was introduced by Hochreiter and Schmidhuber, was designed explicitly for long-term dependencies problem. When predicting future stock values, the system needs to know stock data from a long time ago for analysis, thus it is perfect for the project. Figure 6 displays a diagram of LSTM [9].

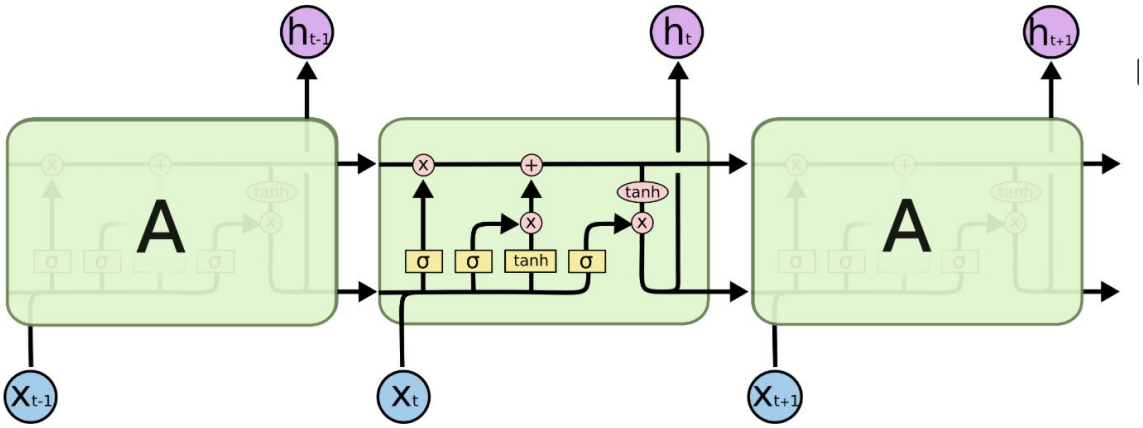

Figure 6. A diagram of long short term memory [9]

The LSTM model uses a more complex algorithm to store previous information than the RNN model. The horizontal line running through the top of the diagram represents the cell state. The cell state runs through the entire system and it stores both long term and short term memories. There are three gates, which lets information go through the cell state, connected to the cell state. The first sigmoid gate determines which information is going to be thrown away by the cell state. It takes in the output from a previous state as well as the input x and produces a number between zero and one based on the inputs. Zero means to throw away the information and one means to let information go through the cell state. The next gate, which includes a sigmoid layer and a tanh layer, decides which new information will be stored in the cell state. This gate consists of two parts. The first part, a sigmoid layer, also known as the input gate layer, checks which values should be updated. Then, a tanh layer generates a candidate for new values that could be added to the cell state. The two parts are combined to create an updated value for the cell state. Finally, the last gate decides which value to output. The next tanh layer will take in the cell state and the output of the sigmoid gate and multiply them together to generate the filtered version of the actual output. This model produces an output based on previous inputs including the ones that were from a while back, preventing the long-term dependencies problem.

When running the LSTM, the dataset should be split into three different sets for generating outputs. The sets include a training set, test set, and validation set. The training set contains a sample of data used to fit the model [10]. The model uses this data to make a prediction. The test set is for evaluating a final model fit on the training set [10]. Lastly, the validation is for evaluating a final model fit while tuning model hyperparameters. The ideal dataset split ratio is 60% for the training set, 20% for the test set, and 20% for the validation set.

3.4.2 Optimizations

Stock market prediction data comes in large volumes. This means that computation times can grow exponentially as we gather more and more data. In order to prevent high wait times for the users, we must make effective optimizations. One way of significantly reducing computation times is by making use of a Graphics Processing Unit (GPU). A GPU allows the computation of multiple units at once. This significantly improves processing time. Without a GPU, the large computation time may lead to the users thinking that the application is “stuck”. This means that it is crucial to reduce the time required to compute prediction values.

Another common approach to performing optimizations is the stochastic gradient descent algorithm. Instead of using this exact algorithm, we use a similar one called Adam Optimization [17]. The Adam Optimization algorithm is a good choice for large datasets that have noise. To begin this algorithm, we first need to define some initial values for the machine learning parameters in the code. These machine learning parameters include: number of unrollings, batch size, size of nodes, and dropout value. A small learning rate value can result in multiple iterations to converge, whereas a large learning rate value can result in overshooting [17]. The learning rate can be changed accordingly throughout the process. Initially, the parameters should be varied individually to determine a common pattern. The runtime should also be considered when making these parameter value changes. After common patterns have been defined for the parameters, multiple parameters can be varied at once. This process continues until the mean squared error (MSE) and average loss is minimized. The values that minimize the MSE and average loss result in the optimal parameter values.

3.4.3 Performance Evaluation

As the machine learning module goes through optimizations, with the use of hardware technologies and data science techniques, we will adopt the trial and error approach to evaluate

performance and to compare trade-offs. Calculations like MSE and average loss are generated to measure algorithm accuracy while algorithm runtime is tracked to ensure that trade-offs are accessed while we improve accuracy. Patterns and trends are detected by varying various parameters of the neural network and are summarized in Table 8.

Table 8. Algorithm response to change in parameter values

Varied Parameter	**MSE Loss**	**Average Loss**	**Runtime**
1) Decreased batch size	Decrease	Decrease	Increase
2) Decreased dropout value	Decrease	Decrease	Increase
3) Decreased number of nodes	Decrease	Decrease	Increase
4) Decreased number of layers(for a total of 200 nodes)	Decrease	Decrease	Decrease
5) Decreased number of layers(for a total of 1500 nodes)	Increase	Decrease	Increase
6) Decreased number of unrollings	Increase	Increase	Decrease

By isolating the variance to individual parameters, we can see that there is a general consistency that improved accuracy comes from increased runtime. Inconsistency in theory is also spotted when comparing the effect of number of layers; the results vary for different quantity of node numbers (see Table 8). Further analysis and research will be conducted to compare the trade-offs in percent accuracy in a multidimensional aspect. All parameters will be varied at the same time to ensure we choose a suitable version of the algorithm that is able to predict stock trends within a reasonable range of accuracy and at the same time meeting our time and resource constraints.

3.4.4 Alternative Algorithms

While LSTM is ideal for making predictions over a long period of time, there are a number of alternative machine learning algorithms that can be used for the same purpose.

3.4.4.1 Moving Average

In the moving average method, the predicted value is an average of the previous stock prices for some arbitrary window size [18]. Each subsequent step takes into consideration the most recent value along and eliminates the oldest value in the set, as expressed in equation below:

$$x_{t+1} = 1/N \sum_{i=t-N}^{t} x_i$$

This technique is adequate for short-term predictions but becomes unreliable when trying to predict for longer periods of time.

3.4.4.2 Linear Regression

Linear regression is an algorithm that uses the relationship between independent variables and the dependent variable to make predictions [19]. For example, a simple linear regression model can have time as the independent variable and the price of the stock as the dependent variable. This model evaluates how the price of a stock changes in accordance with time and predicts a linear future trend based off of previously observed data. Linear regression is excellent for evaluating the effect of individual parameters but because economic data typically is not linear and can have multiple trends, it can be lacking when it comes to time-series forecasting.

3.4.4.3 ARIMA

The Autoregressive Integrated Moving Average model (ARIMA) is a forecasting technique that uses a set of parameters calculated from the dataset to make predictions about the future [20]. These parameters are:

- p: past values for forecasting the next value
- q: past forecast errors to predict future values
- d: order differencing

Tuning of these parameters allows for the most optimal combination of values that minimizes error. ARIMA utilizes concepts from both moving average and linear regression methods in that deviations between consecutive time points and overall patterns in the data are accounted for. In general, ARIMA model parameters are tuned to fit an individual time series and so forecasting another series under the same model may not yield favourable results [21]. For this reason, ARIMA is better for smaller datasets while LSTM works better when dealing with large amounts of data.

4 Data Visualization Design

4.1 Data Visualization Platform

With reliable predictions of stock trends, the results can then be displayed to end-users. It is expected that most of the team members will be contributing to the visual design process so collaborative methods and accessibility are emphasized. Popular visualization tools are selected and compared [22] to meet the needs of this project as shown in Table 8.

Table 9. Basic collaborative requirements for the team

		QlikSense	Power Bi	Tableau
Collaboration	Chats			x
	Ease-of-use	x	x	x
Access	Mac			x
	Windows	x	x	x
Export	Excel/CSV	x	x	x
Import	Excel/CSV	x	x	x
	Amazon S3	x	x	x

For the basic importing and exporting abilities, all three tools were able to meet the requirements of this project--compatibility with Excel, CSV, and Amazon S3. Given that most of the team members come from a technical background, there does not seem to be a challenge in using the visualization tools. However, since the majority of the team members use a Mac environment, Tableau is chosen for its capability to be hosted on Mac.

4.2 User-centered Design

The final goal is to have an all-in-one platform featuring stock information, recommendation, and prediction. With a reliable data source set up, developers can focus on display of information. The UCD(User Centered Design) process will be evaluated throughout the iterative process of design revisions(see Figure 7). Graphs, tables, descriptions, as well as interactive features will be drafted and user surveys will be conducted to under user requirements.

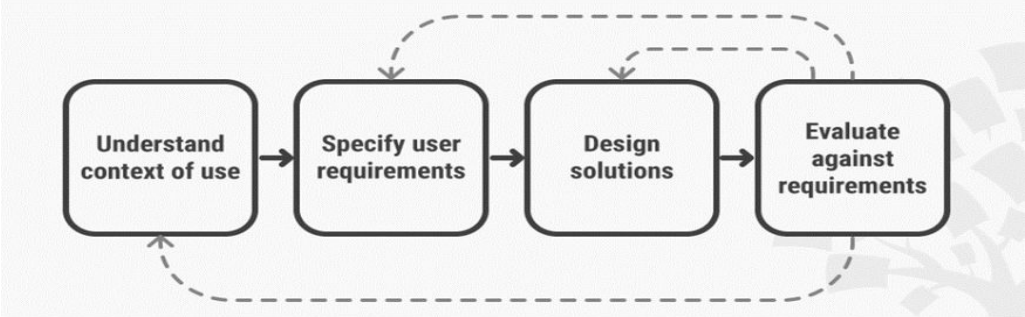

Figure 7. User centered design process [23].

5 Discussion and Project Timeline

5.1 Evaluation of Final Design

The final design meets the project objective and specifications listed in the project specification document. The implementation of user-centred design methods, optimization strategies, machine learning concepts, and cloud technologies creates an all-in-one stock information platform. While designing the stock prediction system, careful consideration was placed upon the weights of various parameters affecting prediction accuracy to generate a more robust and reliable system. This effort to accommodate for all functional and non-functional specifications, along with a well-structured database allows for the design to meet all of the original design specifications.

5.2 Use of Advanced Knowledge

The design and implementation of this project use knowledge from a variety of upper-year engineering courses, including but not limited to two main subject areas: ECE 356 Database Systems and ECE 457A Co-operative and Adaptive Algorithms. To store each stock's price information along with prediction data, knowledge from database subsystems are used to maintain structure. It is crucial that each dataset is organized in the cloud service provider for easy-access and manipulation. The data prediction subsystem borrows concepts from ECE 457A to optimize prediction accuracy. The prediction and recommendation platform utilizes an adaptive recurrent neural network to obtain the most optimal solution.

5.3 Creativity, Novelty, Elegance

The novelty of this design is that it combines a variety of elements from traditional stock market applications to provide an all-in-one platform for stock information, recommendation, and prediction. Where traditional stock market applications all have distinct purposes, this project is ideal for individuals who want a comprehensive overview of the present and future value of their investments. Traditional stock tracking applications allow the user to build a portfolio of real-time stock quotes and track this information. Stock prediction applications typically provide a stock's most likely high, low, opening, and closing prices for a short period of time. This project aims to combine these features in a creative and element manner to best predict stock price patterns over an extended period of time. One of the most elegant features of this project is its user-centred design. From a user-interface perspective to the prediction algorithm, all aspects of this project are centred around information most important to the user.

5.4 Student Hours

Table 10. Total number of hours contributed

Group Member	ID	Hours
Youjing Li	20602178	65
Iruj Fatima	20608774	72
Shumeng Zhang	20621538	60
Seyeon Kim	20614712	70

5.5 Potential Safety Hazards

This project is a pure software project and thus there are no real physical safety hazards associated with its design and implementation.

5.6 Project Timeline

Figures 8 and 9 show the project timeline for Spring 2019 and Winter 2020 respectively.

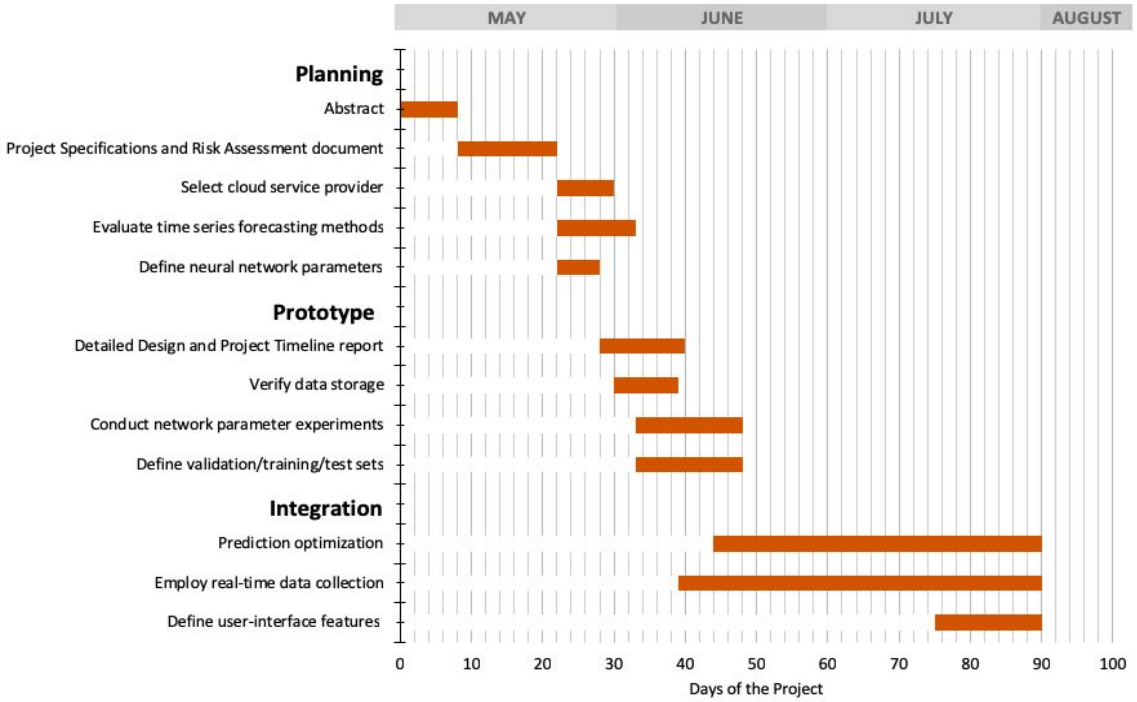

Figure 8. Spring 2019 ECE 498A project timeline

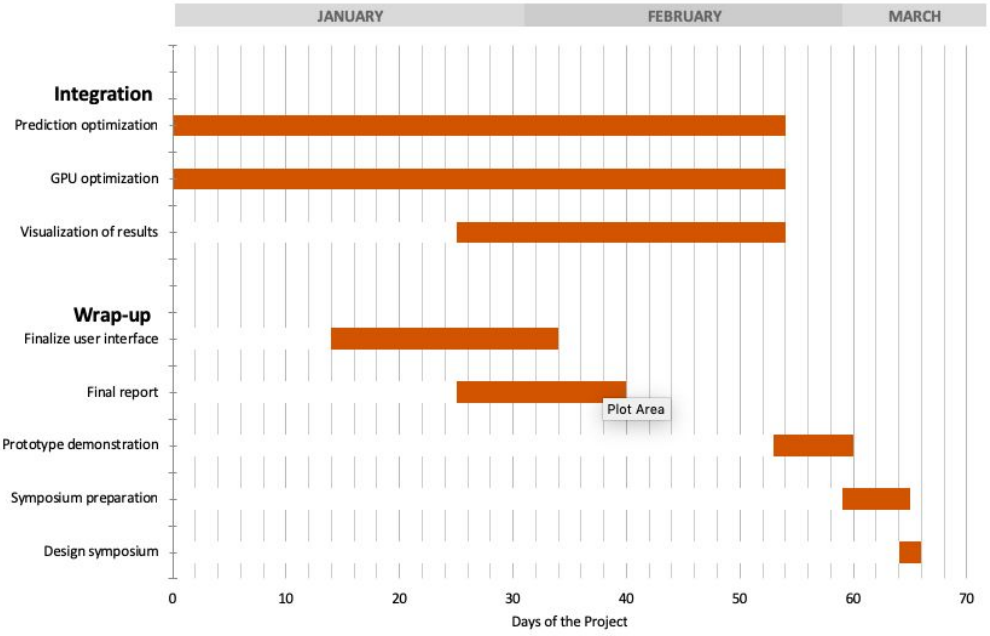

Figure 9. Winter 2020 ECE 493B project timeline

References

[1] Jon Hembrey, "Investing for Beginners: What you need to know", CBC News, 2012. [Online]. Available:https://www.cbc.ca/news/business/taxes/investing-for-beginners-what-you-need-to-know-1.1192836/. [Accessed: 23-May-2019].

[2] C. Voskoglou, “What is the best programming language for Machine Learning?,” *Towards Data Science*, 05-May-2017. [Online]. Available: https://towardsdatascience.com/what-is-the-best-programming-language-for-machine-learning-a745c156d6b7. [Accessed: 25-Jun-2019].

[3] “KDnuggets,” *KDnuggets Analytics Big Data Data Mining and Data Science*, Nov-2017. [Online]. Available: https://www.kdnuggets.com/2017/11/choosing-open-source-machine-learning-library.html. [Accessed: 25-Jun-2019].

[4] “Top 5 best Programming Languages for Artificial Intelligence field,” *GeeksforGeeks*, 15-Oct-2018. [Online]. Available: https://www.geeksforgeeks.org/top-5-best-programming-languages-for-artificial-intelligence-field/. [Accessed: 25-Jun-2019].

[5] “TensorFlow Pros and Cons - The Bright and the Dark Sides,” *DataFlair*, 15-Sep-2018. [Online]. Available: https://data-flair.training/blogs/tensorflow-pros-and-cons/. [Accessed: 25-Jun-2019].

[6] “Pandas Tutorial: DataFrames in Python,” *DataCamp Community*. [Online]. Available: https://www.datacamp.com/community/tutorials/pandas-tutorial-dataframe-python. [Accessed: 25-Jun-2019].

[7] “NumPy,” *NumPy*. [Online]. Available: https://www.numpy.org/. [Accessed: 25-Jun-2019].

[8] “iexfinance,” *PyPI*. [Online]. Available: https://pypi.org/project/iexfinance/. [Accessed: 25-Jun-2019].

[9] “Understanding LSTM Networks,” *Understanding LSTM Networks -- colah's blog*. [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed: 25-Jun-2019].

[10] “About Train, Validation and Test Sets in Machine Learning,” *Towards Data Science*, 06-Dec-2017. [Online]. Available: https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7. [Accessed: 25-Jun-2019].

[11] Amazon Web Services, “Amazon S3 Pricing”, 2019. [Online]. Available: https://aws.amazon.com/s3/pricing/. [Accessed: 21-Jun-2019].

[12] Microsoft Azure, “Blob Storage pricing”, 2019. [Online]. Available: https://azure.microsoft.com/en-ca/pricing/details/storage/blobs/. [Accessed: 22-Jun-2019].

[13] Google Cloud, “Cloud storage pricing”, 2019. [Online]. Available: https://cloud.google.com/storage/pricing. [Accessed: 24-Jun-2019].

[14] Amazon Web Services, “AWS Developer Center”, 2019. [Online]. Available: https://aws.amazon.com/developer/. [Accessed: 24-Jun-2019]

[15] Microsoft Azure, “Supported languages in Azure Functions”, 2019. [Online]. Available: https://docs.microsoft.com/en-us/azure/azure-functions/supported-languages. [Accessed: 42-Jun-2019].

[16] Google Cloud, “Writing Cloud Functions”, 2019. [Online]. Available: https://cloud.google.com/functions/docs/writing/. [Accessed: 24-Jun-2019].

[17] Brownlee, Jason, “Gentle Introduction to the Adam Optimization Algorithm for Deep Learning”, 2017. [Online]. Available: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/. [Accessed: 24-Jun-2019].

[18] Y. Ng, "Machine Learning Techniques applied to Stock Price Prediction", *Towards Data Science*, 2019. [Online]. Available: https://towardsdatascience.com/machine-learning-techniques-applied-to-stock-price-prediction-6c1994da8001. [Accessed: 25- Jun- 2019].

[19] A. Gellert, "Linear Regression Forecasting Method by Companies", *Smallbusiness.chron.com*, 2019. [Online]. Available: https://smallbusiness.chron.com/linear-regression-forecasting-method-companies-73112.html. [Accessed: 25- Jun- 2019].

[20] D. Abugaber, "Chapter 23: Using ARIMA for Time Series Analysis", *Ademos.people.uic.edu*, 2019. [Online]. Available: https://ademos.people.uic.edu/Chapter23.html. [Accessed: 25- Jun- 2019].

[21] G. Box, *Short-Term Forecasting with ARIMA Time Series Models*. 2019

[22] H. Bansal, "Tableau vs Qlik Sense vs Power BI — Choose best BI Tool for Big Data Visualization", *medium.com*, 2019. [Online]. Available: https://medium.com/javarevisited/tableau-vs-qlik-sense-vs-power-bi-choose-best-bi-tool-for-big-data-visualization-533976324c47. [Accessed: 24- Jun- 2019].

[23] Interaction Design Foundation, "User Centered Design", *interaction-design.org*, 2019. [Online]. Available: https://www.interaction-design.org/literature/topics/user-centered-design. [Accessed: 24- Jun- 2019].