SOCH COLLEGE OF IT

AFFILATED TO

TRIBHUVAN UNIVERSITY

INSTITUTE OF SCIENCE AND TECHNOLOGY

Project Report on

NEPSE Stock Forecast

A Stock Price Prediction Application

Submitted To:

Soch College Of IT

(Affiliated to Tribhuvan University)

Ranipawa-11, Pokhara

In partial fulfillment of the requirement for the Bachelor Degree in Computer Science and Information Technology

Submitted By

Kushal Subedi (25085/076)

Niruta Bhattarai (25087/076)

Rachana Regmi (25089/076)

Under the supervision of

Ms.Pratima Baral

March 4,2024

ABSTRACT

In the ever-evolving landscape of finance, predicting stock prices remains a complex challenge. Recognizing this, we introduce a Stock Price Prediction Application tailored for investors. This intuitive platform employs advanced algorithms to forecast stock movements reliably. The user-friendly interface offers insights and predictions, facilitating informed decision-making. By transforming stock market forecasting, our application aims to be a centralized and efficient tool for investors navigating financial intricacies

Keywords: *Stock Price Prediction, Machine Learning, LSTM, Forecasting, Investors, Stock Market, Predictive Model, Stock Movements*

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

AI	Artificial Intelligence
ANN	Artificial Neural Network
API	Application Programming Interface
CNN	Convolutional Neural Network
LSTM	Long Short Term Memory
ML	Machine Learning
NLP	Natural Language Processing
NN	Neural Network
NEPSE	Nepal Stock Exchange
RNN	Recurrent Neural Network
SVM	Support Vector Machine

CHAPTER 1
INTRODUCTION

1.1 Overview

In today's dynamic financial landscape, capital markets provide individuals with a compelling avenue to actively engage in economic activities, especially through investments. Among the various investment options available, stocks emerge as a prominent choice, representing ownership in companies. The returns realized by stockholders are intricately tied to the financial performance of these entities, creating a straightforward equation: when companies thrive, so do their stockholders. However, the promise of significant profits in the stock market is accompanied by greater investment risk. This underscores the increasing importance of predicting future stock prices based on past data. This proposal aims to leverage the capabilities of machine learning to address this critical issue.[1]

The price movements of stocks are not isolated events; they are deeply intertwined with the broader economic landscape. Factors such as monetary policy, influenced by variables like the amount of money in circulation and interest rates, as well as fiscal policy, including taxation policies, play pivotal roles in shaping the behaviour of stock investors.[2]

In the not-so-distant past, various methodologies such as linear regression, time-series analysis, and even chaos theory have been applied to predict stock prices. Yet, with the advent of machine learning, we stand at the threshold of a transformative era in stock price prediction. This project explores how cutting-edge machine learning algorithms and techniques can revolutionise our ability to forecast stock prices accurately, empowering investors with invaluable insights and enabling them to make informed decisions in an ever-evolving financial ecosystem.

1.2 Problem Statement

The challenge at hand is to develop an accurate and reliable method for predicting future stock prices in a volatile financial market. The allure of potentially high returns in the stock market is counterbalanced by the inherent risks, making it imperative to find a robust solution that leverages historical data and machine learning techniques to make informed predictions.

The primary challenge is the development of an accurate and precise stock price prediction

model that can effectively capture the complex interplay of market factors, including economic indicators, geopolitical events, corporate earnings, and investor sentiment.[3]

Moreover, it's widely recognized that managing risk plays a crucial role in investing. With the stock market being so unpredictable, investors need tools to understand and reduce risks effectively. This involves thorough testing and refining the model to make it more accurate and reliable. Dealing with real-time predictions means considering factors like data delays, improving the model continuously, and strategies for using the latest information. Also, it's important to provide user-friendly charts and reports to help investors understand and act on the predictions. To follow the rules and regulations, we make sure we're in line with financial laws. In the end, our approach aims to help investors make well-informed decisions and manage risks in the ever-changing world of financial markets.

1.3 Objectives

* To develop analytical tools for market insights, real-time strategy adjustments, and robust risk assessment.
* To Create accurate predictive models using AI and continuous improvement mechanisms for reliable and transparent stock investment forecasts.
* Enable dynamic scenario analyses, provide market unpredictability education, and implement adaptive strategies for changing market conditions.

1.4 Scope and Limitations

1.4.1 Scope

* **Web Based platform**:Our application will be accessible via the internet, providing a web-based platform.
* **Educational Component**: In addition to prediction, the application may offer educational resources and insights to help users understand the factors affecting stock prices and improve their investment knowledge.
* **Advanced Prediction Models**: Continuous improvement and integration of advanced prediction models to enhance the accuracy of stock price forecasts.

1.4.2 Limitations

- **Accuracy**: The accuracy of the proposed system is highly dependent on that of algorithm used and Quality of Data.
- **Market Volatility**: The stock market is subject to high volatility, and the application may not be able to account for extreme market conditions or sudden economic changes that can significantly affect stock prices.
- **Historical Data Dependency**:The accuracy of predictions heavily relies on historical data. If there are data gaps or inaccuracies in the historical NEPSE data, it may impact the reliability of the predictions.

1.5 Motivation

Stock price prediction is a longstanding and crucial problem. Crafting a dependable model for predicting stock prices can provide valuable insights into how the market behaves over extended periods, helping us identify trends that might otherwise go unnoticed. With the increasing computing power of today's machines, machine learning emerges as an effective approach to address this challenge. Consequently, our inspiration arises from the aspiration to establish a public service that harnesses historical data and user predictions to construct a more resilient model, ultimately benefiting a broader audience.

1.6 Development Methodology

A system development model, also known as a software development life cycle (SDLC) model, is a framework that outlines the processes involved in the planning, creation, testing, deployment, and maintenance of a software system. The purpose of these models is to provide a structured approach to guide the development process, ensuring that software projects are well-organized, efficient, and result in high-quality products.

1.6.1 Agile Model

The Agile model is an iterative and incremental approach to software development. It emphasizes collaboration, flexibility, and customer feedback throughout the development process.

The Agile model is well-suited for our stock prediction application due to the following reasons:

* **User-Centric Approach**: Agile places a strong emphasis on customer feedback and involvement
* **Flexibility**: Agile allows for changes and adaptations as the project progresses. This flexibility is valuable for a stock prediction application because financial markets are dynamic and frequently updating.
* **Research and Development**: Agile encourages a culture of continuous improvement. Which encourages our team to regularly reflect on their processes and seek ways to enhance efficiency and with continious improvement approach.[4]

Figure 1.1: Agile Development

During our project development, we adopted the Agile methodology, specifically leveraging the Scrum framework. Scrum, named after the rugby formation, is a framework designed to facilitate teamwork. Similar to a rugby team preparing for a match, Scrum encourages teams to learn from experiences, self-organize while tackling problems, and reflect on both successes and setbacks for continuous improvement. In line with the Scrum framework, we organized our project into sprints, each being a time-boxed effort with a predefined duration. Typically, the duration ranged from one week to one month, with two weeks being the most common. Throughout the development process, we conducted several collaborative sessions and utilized Trello as our chosen project management tool. Trello is a versatile software that helps software teams plan, track, and manage their work by breaking it down into manageable tasks. With Trello, we seamlessly created user stories, managed issues, planned sprints, and efficiently distributed tasks across our software team. This approach facilitated a systematic development process for our project, encouraging seamless collaboration with field experts and our project supervisor.

1.7 Organization of the Report

This report is organized into six chapters.

First Chapter introduces the project, outlining the problem statement, objectives, scope, and limitations. It also provides an overview of the development methodology and the motivation behind the project.

Second Chapter presents a comprehensive review of the literature, exploring the various methodologies and techniques used in stock price prediction.

Third Chapter delves into the methodology, detailing the approach and techniques used in the development of the stock price prediction model.

Fourth Chapter provides an in-depth analysis of the system architecture, outlining the system's components and their interactions.

Fifth Chapter discusses the implementation of the stock price prediction model, detailing the tools and technologies used in the development process.

Sixth Chapter concludes the report, summarizing the findings and discussing the implications of the stock price prediction model.

CHAPTER 2
BACKGROUND STUDY AND LITERATURE REVIEW

2.1 Background Study

Stock price prediction has been a longstanding challenge in financial markets. Accurate predictions are essential for investors, traders, and financial institutions to make informed decisions. Machine learning has gained significant attention in recent years as a promising approach for improving the accuracy of stock price predictions.[5]. In the early days of trying to predict stock prices, people mainly used traditional math and statistics. They looked at stock charts, used a method called time series analysis, and some math tools like moving averages. Then, in the 1960s and 70s, a person named Benoit Mandelbrot came up with the idea of using something called ”fractal analysis” to understand how stock prices move. But it’s important to know that these early methods had problems because they couldn’t really figure out the complicated and always-changing nature of the stock market.[6] Machine learning algorithms have brought a fresh perspective to research on predicting stock prices. Studies have demonstrated several methods for forecasting stock prices as a result Some of them are below.

Neural Networks:Neural networks have gained popularity in recent years for various machine learning tasks, including stock price prediction [7]. Some neural network algorithms, such as Recurrent Neural Networks (RNNs) and Artificial Neural Networks (ANNs), are commonly practiced for predicting stock prices.

Random Forest Algorithm: Random Forest is a machine learning algorithm introduced by Leo Breiman in 2001 [8]. It is an ensemble learning method that combines the predictions of multiple decision trees to improve accuracy and reduce overfitting.

SVM:In the initial phases of applying machine learning to predict stock prices, the Support Vector Machine (SVM) was utilized. It employed the Support Vector Regression algorithm [9]

2.2 Literature Review

Literature survey is the process in which a complete and comprehensive review is conducted encompassing both the published and unpublished work from other alternative sources of information. This review is conducted in the domains of specific interest to the person or researcher. Further, the results of this process are documented.

”Accurately predicting the change of stock price can reduce the investment risk of stock investors and effectively improve the investment return”. [10] Because of the stock market’s inherent volatility, predicting stock prices typically involves complex nonlinear time series forecasting. Stock prices are influenced by a multitude of variables, making accurate prediction challenging with simplistic models.

Another Survey on *”Chaos theory”*, a branch of mathematics and physics, has found intriguing applications in the field of finance, offering a fresh perspective on understanding and predicting market dynamics. This theory posits that complex systems, such as financial markets, are inherently sensitive to initial conditions and can exhibit chaotic, unpredictable behavior, even though they may appear random at first glance. [11] One of the fundamental concepts of chaos theory is the ”butterfly effect.” It suggests that a small event, like the flapping of a butterfly’s wings, can trigger a cascade of effects that ultimately lead to major, seemingly unrelated events. In the context of finance, this means that minor changes or perturbations in market conditions can have far-reaching consequences. Chaos theory’s recognition of unpredictability and sudden shifts in financial markets underscores the importance of robust risk management strategies. By acknowledging the potential for chaotic behavior, investors and institutions can better prepare for unforeseen market events.

The stock market is significantly impacted by the behavior and trends on social media. Conducting sentiment analysis on social media content can provide valuable insights into trader sentiment and market trends. This, in turn, aids in risk management by allowing for proactive analysis of real-time news and market behavior. [12]

Selecting a model for stock price prediction has become more accessible, despite the dynamic behavior of the stock market. Recurrent Neural Networks (RNN) excel in this context, primarily due to their ability to retain and utilize prior states in current state calculations. Unlike

independent layers, the hidden layers in RNN are interconnected, incorporating not just input layer outputs but also outputs from preceding hidden layers. This characteristic empowers RNN to perform well with sequential data. RNN's advantage lies in its capacity to consider the contextual information within the data during training, making it particularly suitable for stock and Forex scenarios. This is because fluctuations at a specific time often bear a connection to the previous trends. [13]

"Long Short-Term Memory (LSTM) networks have brought a revolutionary shift in the landscape of stock analysis". [7] LSTM networks stand out as the best choice in stock analysis due to their versatility in capturing complex temporal patterns, handling non-linearity, and automatically learning significant features within financial time series data. They empower traders and analysts with an advanced tool for understanding market dynamics and making informed decisions. As research in this field continues to grow, the significance of LSTMs in stock analysis becomes increasingly evident, and their influence in the financial world is set to expand, providing traders with a powerful ally in navigating the intricacies of the stock market. [14]

Study of existing system

On our research we have found various related application on stock price prediction and stock analysis

Chukul NEPSE is an AI-based algorithm Application that aims to predict the trend of Nepal stock market. It is a tool that facilities a trader to make decisions without having to do any technical and fundamental research. They also provide a platform for the user to learn about the stock market and its various aspects. Chukul provides a novice user a safe landing in this unpredictable environment of rumor filled share market. It saves a lot of analysis time regarding the rise and fall of the share value[15].

Nepal Stock Exchange is a mobile Application and web based application that provides updates of daily ups and down in NEPSE. The app provides real-time access to market data, news, and insights pertinent to the Nepal Stock Exchange. The app also provide Comprehensive Analysis of NEPSE Data and news related to Nepal Share Market. Not only that, the app also provides the latest news and updates related to the share market of Nepal. [16]

Saral Lagani is a web based application developed for NEPSE Stock analysis and news pub-

lication. The app provides real-time access to market data, news, and insights pertinent to the Nepal Stock Exchange. This system have tools to analyze stock with interactive graphs and stastical measurements [17]

Yahoo-finance is a web based application that provides stock market data and news. The service covers over 350 exchanges in over 80 countries. Yahoo Finance provides access to more than 5 years of daily OHLC price data. The data can be downloaded in CSV format or accessed through an API. Yahoo Finance also provides access to company news and financials through its website and mobile app. [18]

CHAPTER 3
SYSTEM ANALYSIS

System Analysis is the process of studying a procedure or business in order to identify its goals and purposes and create systems and procedures that will achieve them in an efficient way. It is a problem-solving technique that improves the system and ensures that all components of the system work efficiently to accomplish their purpose. Analysis specifies what the system should do. This Chapter describes the system analysis of the project. It includes the feasibility study, requirement analysis, and system design.

3.1 Requirement Engineering

Based on our background study and literature review (refer to Chapter 2), we've identified several projects similar to ours that employ different methodologies and algorithms. Although our project shares similarities with these endeavors, it's essential to clarify that our primary objective does not revolve around the development of a commercial application. Instead, due to time constraints, we will not be incorporating multiple algorithms.

As a result of this, we have compiled the following set of requirements for our Stock Price Prediction Application.

3.1.1 Functional Requirements

Functional requirements specify the specific functions, features, and capabilities a system or software must have to meet user needs and perform its intended tasks.

i. **Data Collection and Integration:**

- The system should gather historical stock price data from NEPSE .

ii. **Data Preprossing:**

- The system should clean and preprocess raw data, handling missing values and outliers.
- It should normalize or scale data appropriately for modeling.

iii. **Prediction Models:**

- The system should implement LSTM model for prediction .

iv. **Training and Testing:**

- The system should split data into training and testing datasets.
- It should provide options for cross-validation and model evaluation.

v. **Visualization:**

- The system should offer interactive charts and visualizations of historical and predicted stock prices.
- Users should be able to customize the time range and parameters displayed.

3.1.2 Non-Functional Requirements :

i. **Accuracy and Precision:**

- The system should achieve a specified level of accuracy in stock price predictions.
- Precision requirements should be defined to minimize false positives in alerting.

ii. **Scalability:**

- The system should handle a growing volume of data and users without performance degradation.
- It should support parallel processing or distributed computing for scalability.

iii. **Reliability:**

- The system should be available and reliable 24/7 to support real-time trading decisions.
- Redundancy and failover mechanisms should be in place to minimize downtime.

iv. **Response Time:**

- The system should provide real-time or near-real-time predictions and alerts.
- Response time requirements should be specified for different functions.

v. **User Interface:**

- The user interface should be intuitive and user-friendly, catering to both novice and experienced users.
- Accessibility and usability guidelines should be followed.

3.2 Feasibility Study

A feasibility study is a comprehensive analysis conducted to assess the practicality and viability of a proposed project, system, product, or business idea. It serves as a critical tool for decision-making, helping stakeholders determine whether a project should be pursued, modified, or abandoned. The primary objectives of a feasibility study are to evaluate the project's potential for success, identify potential challenges, and provide a basis for informed decision-making.

3.2.1 Technical Feasibility

In the technical aspect of our system, which relies on algorithmic and machine learning models, we will utilize **Python** as our primary programming language. We'll leverage essential libraries such as **Pandas, Numpy, Keras, and scikit-learn** to support our core functionalities. As the system grows and evolves, there may be a need to incorporate additional programming languages like JavaScript, and markup languages like HTML and CSS to facilitate further expansion and enhance the user interface.

3.2.2 Operational Feasibility

For our stock prediction system, operational feasibility means making sure we have the right teammates who understand Python, machine learning, and web technologies. We also need to check that our computers and internet systems can handle the work. We must make sure our data is managed well, and we follow the rules about data privacy. We'll need plans to keep the system running, and handle any future growth. If we can do all this, our stock prediction system will work smoothly and give us the information we need.

3.2.3 Economical Feasibility

The primary expenses associated with developing the system revolve around ensuring that we have adequately equipped devices and a reliable internet connection. Our economic viability

also encompasses costs related to travel, wages for staff during system launch, as well as expenses for meals. Additionally, it includes spending on online courses and books for research and study purposes. Deployment costs are another aspect of our economic feasibility plan, along with potential expenses for accessing third-party system APIs.

Cost/Revenue Item	**Amount (NRs:)**
Initial Investment	5,000
Internet and Device Costs	25,000
Travel Expenses	4,500
Staff Wages (Launch Phase)	15,000
Research Materials (Courses, Books)	7,500
Deployment Costs	3,000
Third-Party API Expenses	1,200
Total Costs	61,200
Projected Sales/Income	20,000
Break-Even Point (BEP)	41,200

Table 3.1: Break-Even Analysis

3.3 Use Case Diagram

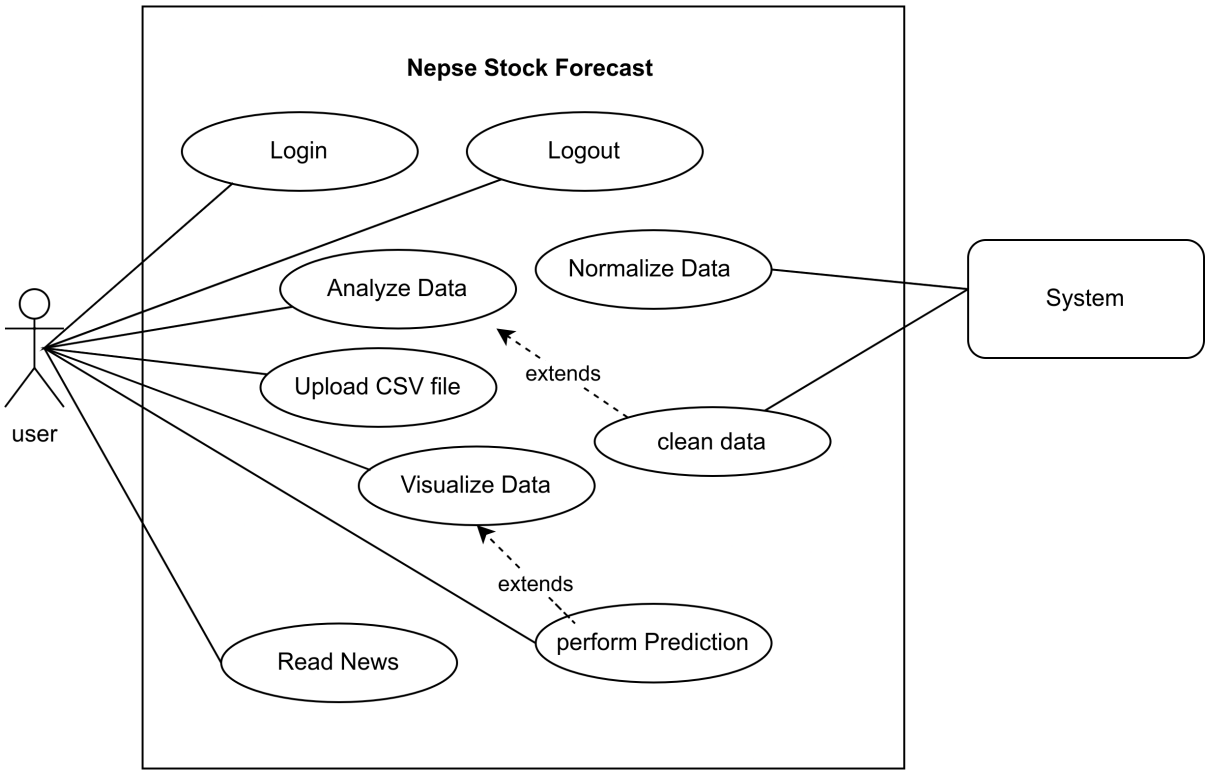

Figure 3.1: Usecase Diagram

The Usecase Diagram shows the user interaction with the system and the system's response to the user's actions. The user can interact with the system by logging in, viewing stock data, and making predictions. The system will respond by providing the user with the requested data and predictions. The system will also provide visualizations of the historical and predicted stock prices.

CHAPTER 4
SYSTEM DESIGN

This chapter gives an overview of the proposed system along with defining the system architecture and UML diagrams such as use-case and class diagrams. A UML diagram is a diagram based on the UML (Unified Modelling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

4.1 Overview of System Design

We are implementing a system that will create a predictive model using a custom training algorithm. Users can make decisions to buy or sell stocks based on the model's predictions. The system will calculate profit or loss based on the closing price of a stock for the day, which we consider as the target variable.

Forecasting stock market behavior can seem complicated because there are many unknown factors that don't seem to follow a straightforward pattern. However, by using machine learning effectively, we can teach computers to recognize trends and make informed guesses by looking at past and current data.

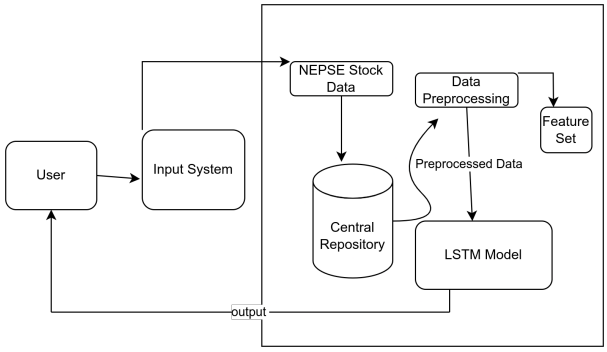

Figure 4.1: System Architecture

4.2 System Flow Diagram

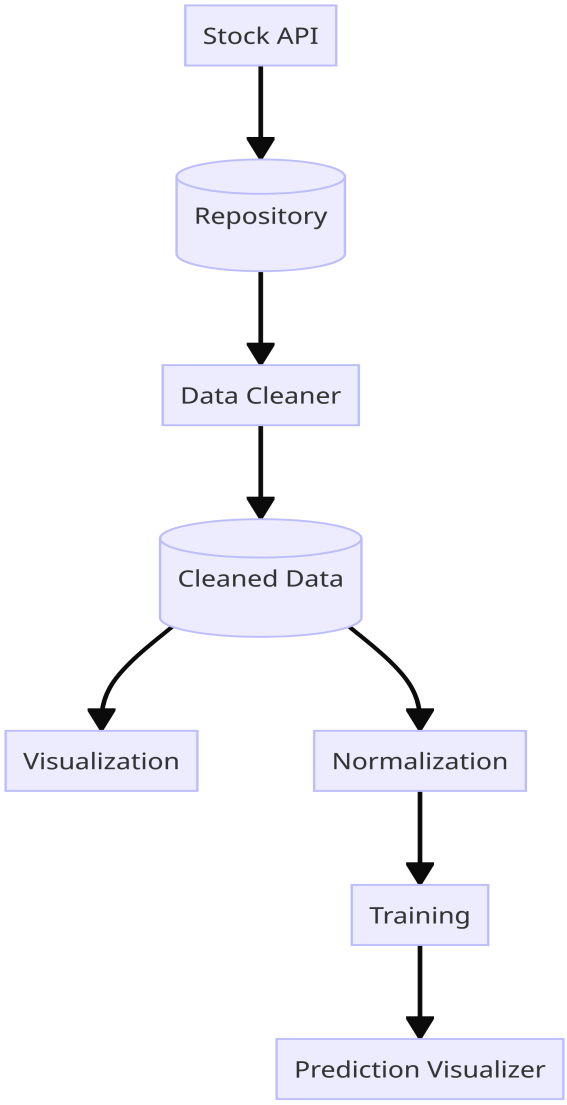

Figure 4.2: System Flow Diagram

The system flow diagram illustrates the process whereby our system retrieves data from stock APIs and stores it in a central repository. Subsequently, the data undergoes cleaning, visualization, normalization, and training stages before predictions are generated.

4.3 Activity Diagram

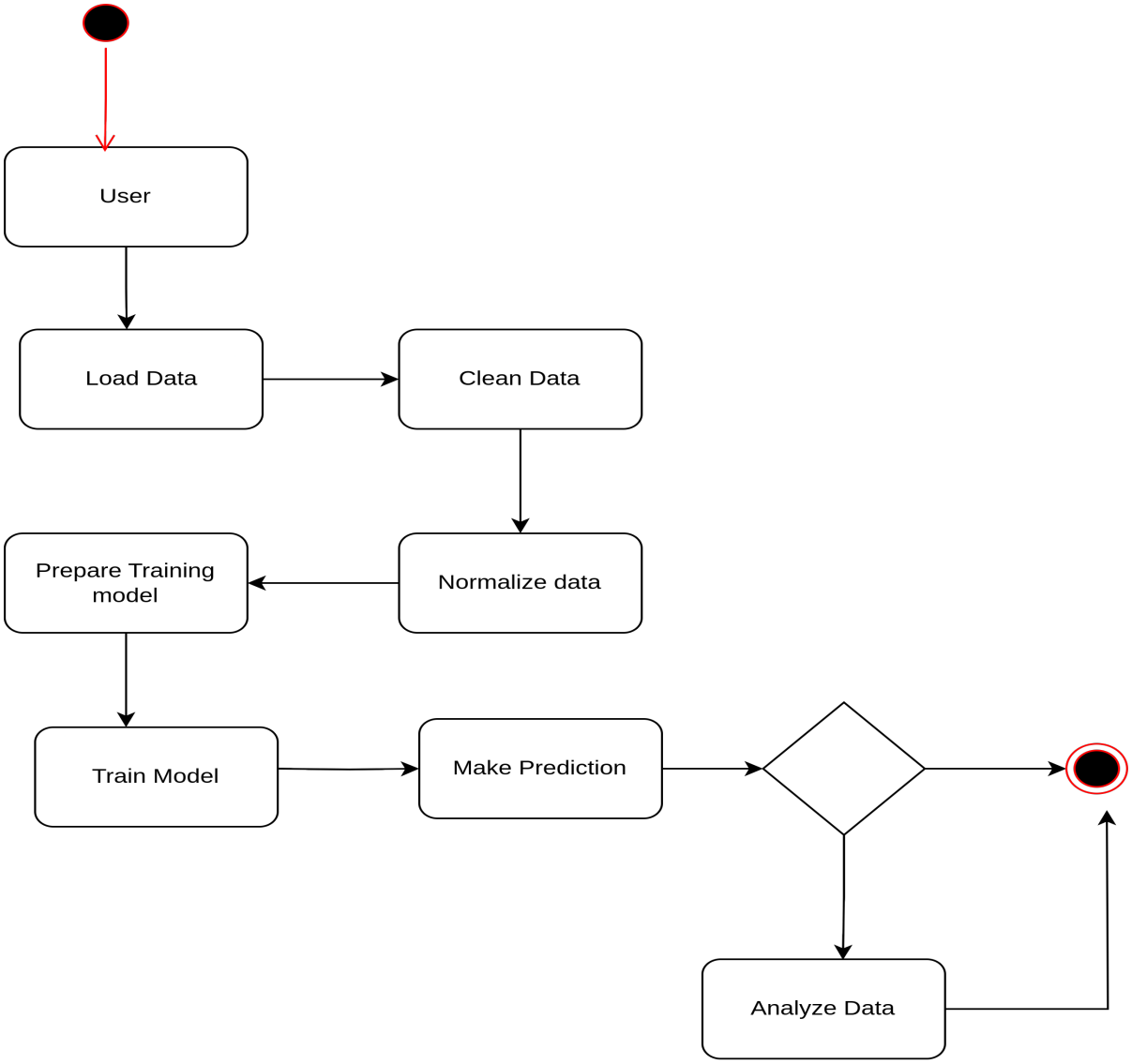

Figure 4.3: Activity Diagram

The activity diagram fot the system shows the process of data collection, preprocessing, model training, and prediction generation. The system will also provide visualizations of the historical and predicted stock prices.

4.4 Class Diagram

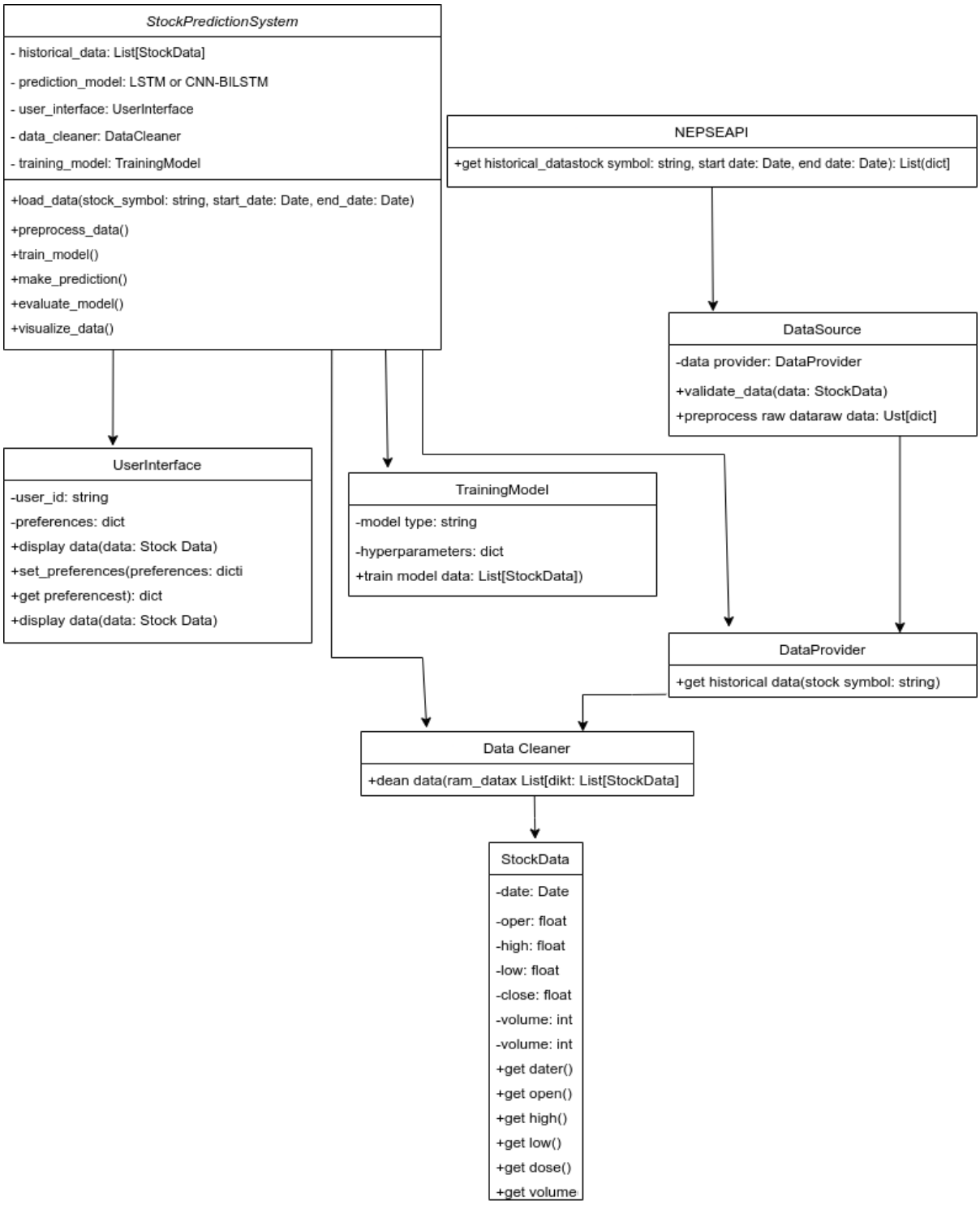

Figure 4.4: Class Diagram

The class diagram delineates the connections and data flow between base and derived classes. Within the system, the Datasource class is responsible for collecting data from the stock API and storing it centrally. The DataCleaner class performs data cleaning and normalization tasks, while the TrainingModel class handles model training and prediction generation. Serving as the foundation for the application, the StockPredictionSystem class manages user functions and interacts with the other classes.

4.5 Algorithm Details

We are using LSTM algorithm for our project. LSTM is a special kind of RNN, which shows outstanding performance on a large variety of problems. LSTM is well-suited to classify, process and predict time series given time lags of unknown duration. It trains the model by using backpropagation. Backpropagation is a method used in artificial neural networks to calculate a gradient that is needed in the calculation of the weights to be used in the network. Backpropagation is shorthand for ”the backward propagation of errors,” since an error is computed at the output and distributed backwards throughout the network’s layers. It is commonly used to train deep neural networks, a term referring to neural networks with more than one hidden layer. [19]

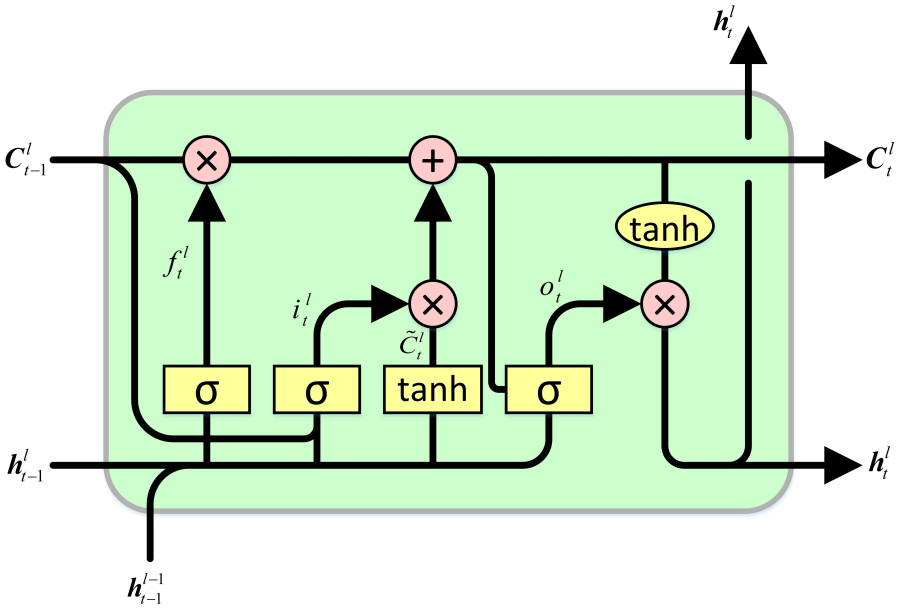

Figure 4.5: LSTM Architecture

Pseudo code

A pseudo code is a detailed yet readable description of what a computer program or algorithm must do, expressed in a formally-styled natural language rather than in a programming language. It is a high-level description of the actions of a program or algorithm, using a mixture of English and informal programming language syntax. Here is the pseudo code for LSTM algorithm.

Initialization

Initialize weights and biases:

$W_ih \leftarrow initialize\_weights(input\_size, hidden\_size)$

$W_hh \leftarrow initialize\_weights(hidden\_size, hidden\_size)$

$b_ih \leftarrow initialize\_biases(hidden\_size)$

$b_hh \leftarrow initialize\_biases(hidden\_size)$

$W_ho \leftarrow initialize\_weights(hidden\_size, output\_size)$

$b_o \leftarrow initialize\_biases(output\_size)$

Initial States

Initialize cell_state and hidden_state:

$cell\_state \leftarrow initialize\_zeros(hidden\_size)$

$hidden\_state \leftarrow initialize\_zeros(hidden\_size)$

Forward Pass

for each $input\_t$ in $input\_sequence$ **do**

Calculate input gate, forget gate, cell state, and output gate:

$i_t \leftarrow sigmoid(W_ih \cdot input_t + W_hh \cdot hidden\_state + b_ih + b_hh)$

$f_t \leftarrow sigmoid(W_ih \cdot input_t + W_hh \cdot hidden\_state + b_ih + b_hh)$

$cell\_state \leftarrow f_t \cdot cell\_state + i_t \cdot tanh(W_ih \cdot input_t + W_hh \cdot hidden\_state + b_ih + b_hh)$

$o_t \leftarrow sigmoid(W_ho \cdot hidden\_state + b_o)$

Update hidden_state:

$hidden\_state \leftarrow o_t \cdot tanh(cell\_state)$

end for

Training

for each $epoch$ in $epochs$ **do**

for each $example, label$ in $zip(train\_data, labels)$ **do**

Perform forward pass to get predicted_output:

$predicted\_output \leftarrow lstm\_forward(example)$

Calculate loss:

$loss \leftarrow compute\_loss(predicted\_output, label)$

Perform backward pass to update weights and biases:

$backward\_pass(loss, learning\_rate)$

end for

end for

CHAPTER 5
IMPLEMENTATION

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective. The implementation stage involves careful planning, investigation of the existing system and it's constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

5.1 Tools and Technologies

5.1.1 Languages and Frameworks

- **Python**: Python is an interpreted, high-level and general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant indentation. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

- **Google Collab**: Google Colab is a free cloud service and now it supports free GPU! You can; improve your Python programming language coding skills. develop deep learning applications using popular libraries such as Keras, TensorFlow, PyTorch, and OpenCV.

- **Scikit Learn**: Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

- **TensorFlow**: TensorFlow is a free and open-source software library for machine learn-

ing. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. Tensorflow is a symbolic math library based on dataflow and differentiable programming.

◇ **Django**: Django is a Python-based free and open-source web framework that follows the model-template-views architectural pattern. It is maintained by the Django Software Foundation. Django's primary goal is to ease the creation of complex, database-driven websites. [20]

5.1.2 CASE Tools

◇ **Draw.io**: Draw.io is a free online diagram software for making flowcharts, process diagrams, org charts, UML, ER and network diagrams.

◇ **Vs-Code**: Visual Studio Code is a free source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.

◇ **Git**: Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

◇ **Trello**: Trello is a web-based Kanban-style list-making application which is a subsidiary of Atlassian.

5.2 Working Of LSTM Algorithm

The Long Short-Term Memory (LSTM) network is a type of recurrent neural network (RNN) designed to address the vanishing gradient problem that occurs in traditional RNNs. Here is a high-level mathematical derivation of the LSTM update equations. Let's consider an LSTM cell at time step t. The LSTM cell maintains a cell state (c_t), hidden state (h_t), input gate (i_t), forget gate (f_t), output gate (o_t), and memory cell content (g_t). The update equations for these components are given by:

1. **Input Gate (i_t):**

$$i_t = \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi})$$

2. **Forget Gate (f_t):**

$$f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf})$$

3. **Output Gate (o_t):**

$$o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho})$$

4. **Cell State Update (g_t):**

$$g_t = \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{t-1} + b_{hg})$$

5. **Cell State (c_t) Update:**

$$c_t = f_t c_{t-1} + i_t g_t$$

6. **Hidden State (h_t) Update:**

$$h_t = o_t \tanh(c_t)$$

In these equations:

- x_t is the input at time step t.
- $W_{ii}, W_{hi}, b_{ii}, b_{hi}$ are weight matrices and bias terms for the input gate.
- $W_{if}, W_{hf}, b_{if}, b_{hf}$ are weight matrices and bias terms for the forget gate.
- $W_{io}, W_{ho}, b_{io}, b_{ho}$ are weight matrices and bias terms for the output gate.
- $W_{ig}, W_{hg}, b_{ig}, b_{hg}$ are weight matrices and bias terms for the cell state update.
- σ represents the sigmoid activation function, and tanh represents the hyperbolic tangent activation function.

These equations describe how information is updated and propagated through the LSTM cell at each time step. The input gate controls how much of the new information should be added to the cell state, the forget gate controls how much of the previous cell state should be retained, and the output gate controls how much of the cell state should be exposed to the next layer. This architecture allows LSTMs to capture and remember long-term dependencies in sequential data, making them suitable for tasks such as natural language processing, time series prediction, and more. [21]

5.2.1 High Level Overview Of LSTM algorithm

1. **Import Libraries:**

```
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense
```

2. **Prepare Sequential Data:**

```
# Example: Generate sequential data
sequence_length = 10
X_train = np.random.rand(100, sequence_length, 1)
y_train = np.random.randint(0, 2, size=(100,))
```

3. **Build LSTM Model:**

```
model = Sequential()
model.add(LSTM(units=50,
    input_shape=(sequence_length, 1)))
model.add(Dense(units=1, activation='sigmoid'))
```

4. **Compile the Model:**

```
model.compile(optimizer='adam',
```

```
        loss='binary_crossentropy', metrics=['accuracy'])
```

5. **Train the Model:**

```
        model.fit(X_train, y_train, epochs=10, batch_size=32)
```

6. **Make Predictions:**

```
        X_test = np.random.rand(10, sequence_length, 1)
        predictions = model.predict(X_test)
```

7. **Evaluate the Model:**

```
        # Example: Generate test data
        X_test = np.random.rand(20, sequence_length, 1)
        y_test = np.random.randint(0, 2, size=(20,))

        # Evaluate the model
        loss, accuracy = model.evaluate(X_test, y_test)
        print(f"Test Loss: {loss}, Test Accuracy: {accuracy}")
```

5.3 Testing

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. Testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

Table 5.1: Algorithm Testing

sn	Stock Symbol	LTP	Actual Opening Price	Predicted Opening Price	Difference
1	NABIL	491.20	497	502	+5
2	ACLBSL	607.00	610	606	-4
3	ADBL	245.10	245	238	-7

Accuracy of the model

The model is trained with NEPSE stock datas and tested with the same data. The 180 days data is divided into training and testing parts in the ratio of 80:20. The model is trained with 80% of the data and tested with 20% of the data. After training, the LSTM algorithm achieved an accuracy of 72% on the testing data, using a threshold of 0.85 for classifying stock movements as ”up” or ”down”. This accuracy indicates the model's ability to make accurate predictions on unseen data. Additionally, performance metrics such as precision, recall, and F1-score can provide further insights into the model's performance and its ability to correctly classify stock movements.

Precision: Precision measures the proportion of true positive predictions among all positive predictions made by the model. It is calculated as the ratio of true positives to the sum of true positives and false positives.

$$Precision = \frac{TP}{TP + FP}$$

False Positive Rate (FPR): FPR measures the proportion of false positive predictions among all actual negative instances in the dataset. It is calculated as the ratio of false positives to the sum of false positives and true negatives.

$$FPR = \frac{FP}{TN + FP}$$

Balanced Accuracy: Balanced accuracy calculates the average of sensitivity and specificity and is suitable for imbalanced datasets.

$$Balanced\ Accuracy = \frac{1}{2} \cdot (Sensitivity + Specificity)$$

F1 Score: The F1 score is the harmonic mean of precision and recall, providing a single metric

that balances both measures. It is calculated as

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Precision: 0.722

F1 Score: 0.8387

Accuracy: 0.722

Balanced Accuracy: 0.5

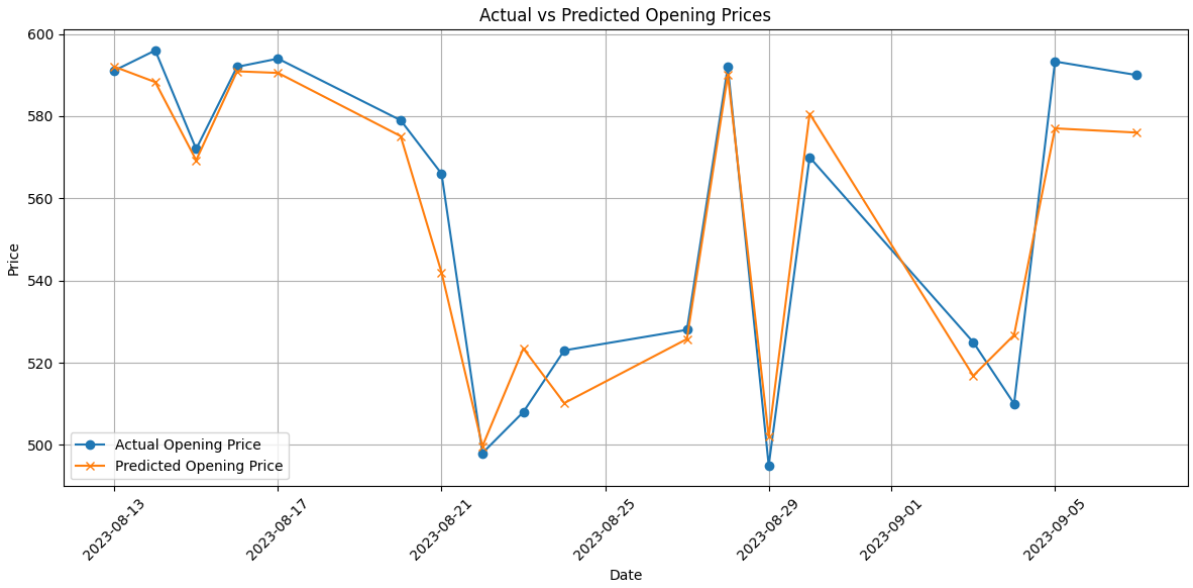

Figure 5.1: Algorithm Testing

5.4 User Acceptance Testing

Table 5.2: User Acceptance Testing

sn	Can Train	Expected Outcome	Can Visualize	Test Result
1	Yes	Yes	Yes	Pass
2	Yes	Yes	Yes	Pass
3	Yes	Yes	Yes	Pass

CHAPTER 6
CONCLUSION AND FUTURE WORK

6.1 Conclusion

In conclusion, the development of a stock price prediction system utilizing a machine learning model, specifically LSTM, holds great promise in enhancing the financial decision-making process. With the expected outcome of this system, investors and analysts will have a robust tool at their disposal, offering accurate stock predictions and risk management capabilities. The extensive research and incorporation of the latest industry trends underscore the commitment to providing users with valuable insights into the stock market's complexities. By empowering users to make informed investment decisions, this system aims to not only improve profitability but also mitigate financial risks. In an ever-evolving financial landscape, such a system can be a valuable asset for those seeking to navigate the stock market with greater confidence and effectiveness.

6.2 Future Work

The future work for this project includes the following:

- **Enhanced Model Performance:** The LSTM model's performance can be further enhanced by incorporating additional features and data sources. This can include the integration of social media sentiment analysis, news articles, and other relevant data to improve the model's predictive capabilities.
- **Real-time Data Integration:** The system can be further developed to incorporate real-time data feeds, enabling users to access the latest market information and predictions. This can provide users with up-to-the-minute insights and enhance the system's overall utility.
- **Scalability and Performance Optimization:** As the user base and data volume grow, the system's scalability and performance optimization become critical. Future work can focus on enhancing the system's scalability and optimizing its performance to accommodate increasing demands.

REFERENCES

[1] F. Momin, S. Patel, K. Shide, and P. A. C. Taskar, "Stock market prediction system using machine learning approach," 2019, October 2019.

[2] J. L. Ticknor, "A bayesian regularized artificial neural network for stock market forecasting," vol. 40, no. 14, 2013, 15 October 2013.

[3] W. A. Mohammed, "Challenges of stock prediction," in *Valuation Challenges and Solutions in Contemporary Businesses*, IGI Global, 2020, pp. 234–252.

[4] B. Boehm, "A survey of agile development methodologies," *Laurie Williams*, vol. 45, p. 119, 2007. [Online]. Available: https://www.researchgate.net/profile/Barry-Boehm/publication/220626009_A_Survey_of_Agile_Development_Methodologies/links/0fcfd50e6b7d9e0c6c000000/A-Survey-of-Agile-Development-Methodologies.pdf.

[5] J. Chen, Y. Wen, Y. Nanehkaran, M. Suzauddola, W. Chen, and D. Zhang, "Machine learning techniques for stock price prediction and graphic signal recognition," *Engineering Applications of Artificial Intelligence*, vol. 121, p. 106 038, 2023, ISSN: 0952-1976. DOI: https://doi.org/10.1016/j.engappai.2023.106038. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0952197623002221.

[6] B. Mandelbrot, "The variation of certain speculative prices," *Journal of Business*, vol. 36, no. 4, pp. 394–419, 1963.

[7] D. Zhang and S. Lou, "The application research of neural network and bp algorithm in stock price pattern classification and prediction," *Future Generation Computer Systems*, vol. 115, pp. 872–879, 2021, ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2020.10.009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X20329861.

[8] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[9] M. Beniwal, A. Singh, and N. Kumar, "Forecasting long-term stock prices of global indices: A forward-validating genetic algorithm optimization approach for support vector regression," *Applied Soft Computing*, vol. 145, p. 110 566, 2023, ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2023.110566. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494623005847.

[10] W. Lu, J. Li, J. Wang, and et al., "A cnn-bilstm-am method for stock price prediction," *Neural Computing & Applications*, vol. 33, pp. 4741–4753, 2021. DOI: 10.1007/s00521-020-05532-z. [Online]. Available: https://doi.org/10.1007/s00521-020-05532-z.

[11] I. Klioutchnikov, M. Sigova, and N. Beizerov, "Chaos theory in finance," *Procedia Computer Science*, vol. 119, pp. 368–375, 2017, 6th International Young Scientist Conference on Computational Science, YSC 2017, 01-03 November 2017, Kotka, Finland, ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2017.11.196. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050917324067.

[12] J. Bollen, H. Mao, and X. Zeng, "Twitter mood predicts the stock market," *Journal of Computational Science*, vol. 2, no. 1, pp. 1–8, 2011. DOI: 10.1016/j.jocs.2010.12.007.

[13] Z. Hu, Y. Zhao, and M. Khushi, "A survey of forex and stock price prediction using deep learning," *Applied System Innovation*, vol. 4, 2021, ISSN: 2571-5577. DOI: 10.3390/asi4010009. [Online]. Available: https://www.mdpi.com/2571-5577/4/1/9.

[14] G. Zhang, W. Xie, and Y. Zhu, "Stock price prediction with time lags using a hybrid deep learning framework," *Applied Soft Computing*, vol. 70, pp. 525–533, 2018. DOI: 10.1016/j.asoc.2018.05.042. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S156849461830346X.

[15] Pi Tech Private Limited, *Chukul nepse*, All rights reserved. [Online]. Available: %5Curl%7Bhttps://chukul.com/#/%7D.

[16] NEPSE, *Nepal share*, ALL rights reserved . [Online]. Available: %7Bhttps://www.nepalstock.com/%7D.

[17] Saral Lagani, *Saral lagani*, All rights reserved. [Online]. Available: %5Curl%7Bhttps://sarallagani.com/%7D.

[18] Yahoo Finance, *Yahoo finance*, All rights reserved. [Online]. Available: %5Curl%7Bhttps://finance.yahoo.com/%7D.

[19] I. JAAMU, "What is lstm? introduction to long short term memory," 2023. [Online]. Available: https://intellipaat.com/blog/what-is-lstm/.

[20] Python, *Python*, All rights reserved. [Online]. Available: %5Curl%7Bhttps://www.python.org/%7D.

[21] C. Olah, "Understanding lstm networks," 2015. [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

APPENDIX

Data Scraping Code

```
from bs4 import BeautifulSoup
import requests
import csv
# Site Url
URL = "https://merolagani.com/LatestMarket.aspx"
# make get request to fetch the raw html content
conn = requests.get(URL)
soup = BeautifulSoup(conn.text, 'lxml')
table = soup.find('table', class_='table table-hover live-trading sortable')
headers = [i.text for i in table.find_all('th')]
print(headers)
data = [j for j in table.find_all(
    'tr', {"class": ["decrease-row", "increase-row", "nochange-row"]})]
result = [{headers[index]: cell.text for index,
           cell in enumerate(row.find_all('td'))} for row in data]
# writing the data in CSV file with headers and their respective values
with open('nepse.csv', 'w') as f:
    try:
        writer = csv.DictWriter(f, fieldnames=headers)
        writer.writeheader()
        writer.writerows(result)
        print("CSV file created successfully and data written successfully")
    except Exception as e:
        print(e)
f.close()
```

Symbol Data Scraping Code

```
from selenium import webdriver
from selenium.webdriver.chrome.options import Options
from bs4 import BeautifulSoup
from selenium.webdriver.common.by import By
import time
import csv
StockSymbol = input("Enter the stock symbol: ")
time.sleep(2)
driver = webdriver.Chrome()
url = f"https://merolagani.com/CompanyDetail.aspx?symbol={StockSymbol}#0"
driver.get(url)
print (" ......... Fetching Data From Merolagani ............ ")\
button = driver.find_element(By.XPATH,
    '//*[@id="ctl00_ContentPlaceHolder1_CompanyDetail1_lnkHistoryTab"]'
)
button.click()
time.sleep(2)
page_source = driver.page_source
soup = BeautifulSoup(page_source, 'html.parser')
table = soup.find_all('table', class_='table table-bordered table-striped
   table-hover')
headers = [i.text.replace('\n', '') for i in table[0].find_all('th')]
print(headers)
data = [[i.text for i in row.find_all('td')] for row in table[0].find_all('tr'
   )]
driver.quit()
print(" ......... Generating CSV File OF Company Data ............ ")
time.sleep(2)
try:
    with open(f'{StockSymbol}.csv', 'w', newline='') as file:
```

```
        writer = csv.writer(file)
        writer.writerow(headers)
        writer.writerows(data)
        print("Done")
except Exception as e:
    print(e)
    print("Error in generating CSV file")
```

Stock Price Prediction Code

```
"""
Importing Libraries
"""
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense
import matplotlib.pyplot as plt
"""
Loading CSV Data File
"""
file_path = 'NABIL.csv'
df = pd.read_csv(file_path)
#remove missing valuse
df = df.dropna()
#scaling data using min-max scaler
scaler = MinMaxScaler()
df_scaled = scaler.fit_transform(df[['Open']])
#Create Dataset
def create_dataset(df_scaled, look_back=1):
  x_train = []
  y_train = []
  for i in range(len(df_scaled) - look_back):
    x_train.append(df_scaled[i:i + look_back])
    y_train.append(df_scaled[i + look_back])
  return x_train, y_train
#
sequence_length = 10
X, y = create_dataset(df_scaled, sequence_length)
```

```
train_size = int(len(X) * 0.8)
X_train, X_test = X[:train_size], X[train_size:]
y_train, y_test = y[:train_size], y[train_size:]
model = Sequential()
model.add(LSTM(50, activation='relu', return_sequences=True, input_shape=(
    sequence_length, 1)))
model.add(LSTM(50, activation='relu'))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')
# Convert lists to arrays and reshape for LSTM input
X_train = np.array(X_train).reshape((len(X_train), sequence_length, 1))
X_test = np.array(X_test).reshape((len(X_test), sequence_length, 1))
y_train = np.array(y_train).reshape((len(y_train), 1))
y_test = np.array(y_test).reshape((len(y_test), 1))
model.fit(X_train, y_train, epochs=300, batch_size=32, validation_data=(X_test
    , y_test), verbose=2)
# Make predictions
train_predictions = model.predict(X_train)
test_predictions = model.predict(X_test)
# Inverse transform the predictions to the original scale
train_predictions = scaler.inverse_transform(train_predictions)
y_train_original = scaler.inverse_transform(y_train.reshape(-1, 1))
test_predictions = scaler.inverse_transform(test_predictions)
y_test_original = scaler.inverse_transform(y_test.reshape(-1, 1))
# Plot the results
plt.figure(figsize=(12, 6))
# Plot training data
plt.plot(df.index[sequence_length:sequence_length+train_size],
    y_train_original, label='Actual Train Data')
plt.plot(df.index[sequence_length:sequence_length+train_size],
    train_predictions, label='Train Predictions')
# Plot testing data
```

```
test_index = df.index[sequence_length + train_size:sequence_length +
    train_size + len(test_predictions)]
plt.plot(test_index, y_test_original, label='Actual Test Data')
plt.plot(test_index, test_predictions, label='Test Predictions')
plt.legend()
plt.show()
df = df.sort_index(ascending=False)
# Scale the most recent data
latest_data = df.head(sequence_length)
latest_scaled = scaler.transform(latest_data[['Open']])
# Reshape the scaled data
latest_scaled = np.array(latest_scaled).reshape((1, sequence_length, 1))
# Predict tomorrow's price
tomorrow_prediction = model.predict(latest_scaled)
tomorrow_prediction = scaler.inverse_transform(tomorrow_prediction)
print("Tomorrow's predicted price:", tomorrow_prediction[0, 0])
```

Figures and Graphs

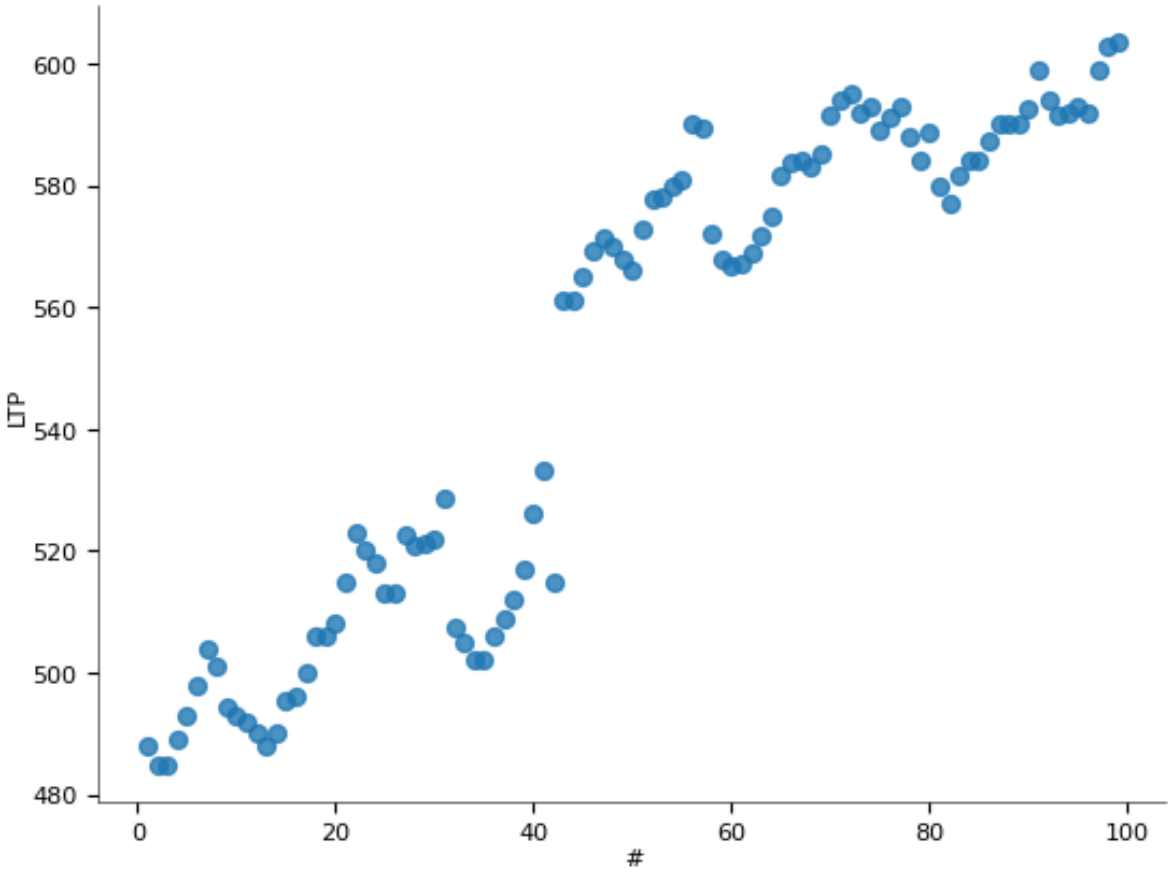

Figure 6.1: Data Distribution

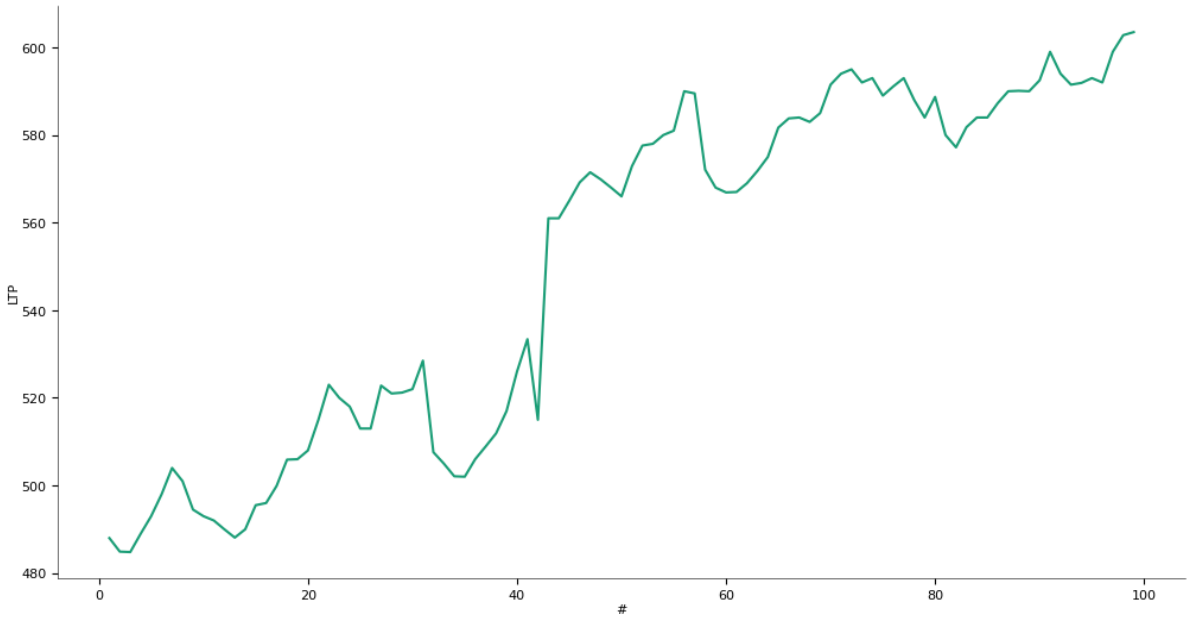

Figure 6.2: Time series of LTP price

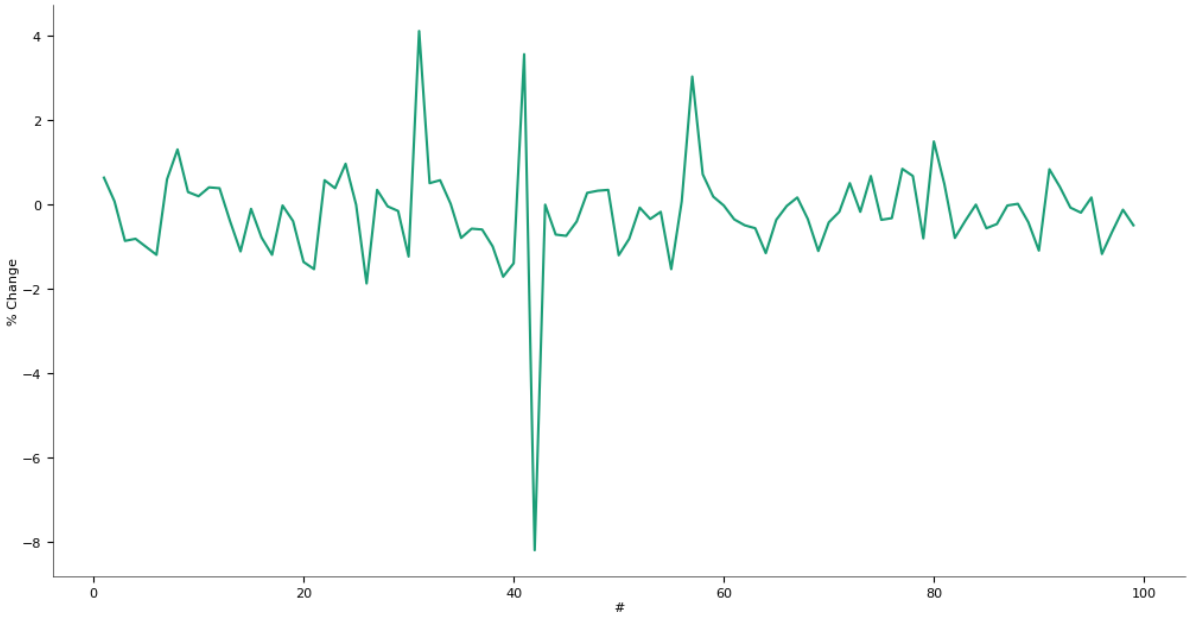

Figure 6.3: Value Change Graph

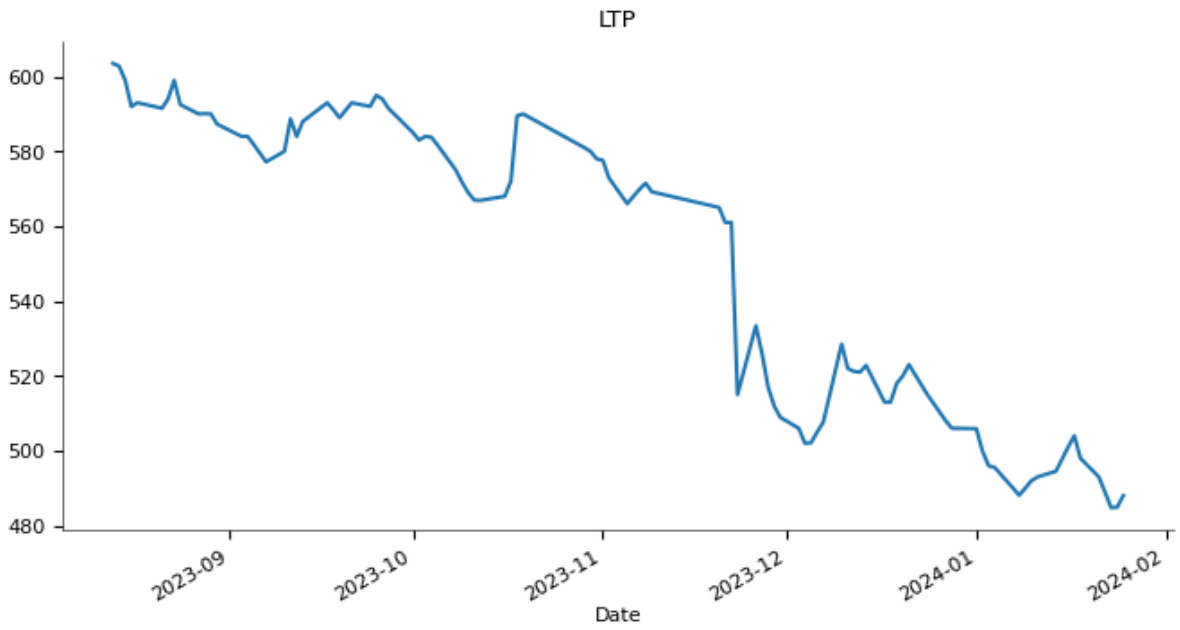

Figure 6.4: Time series of LTP

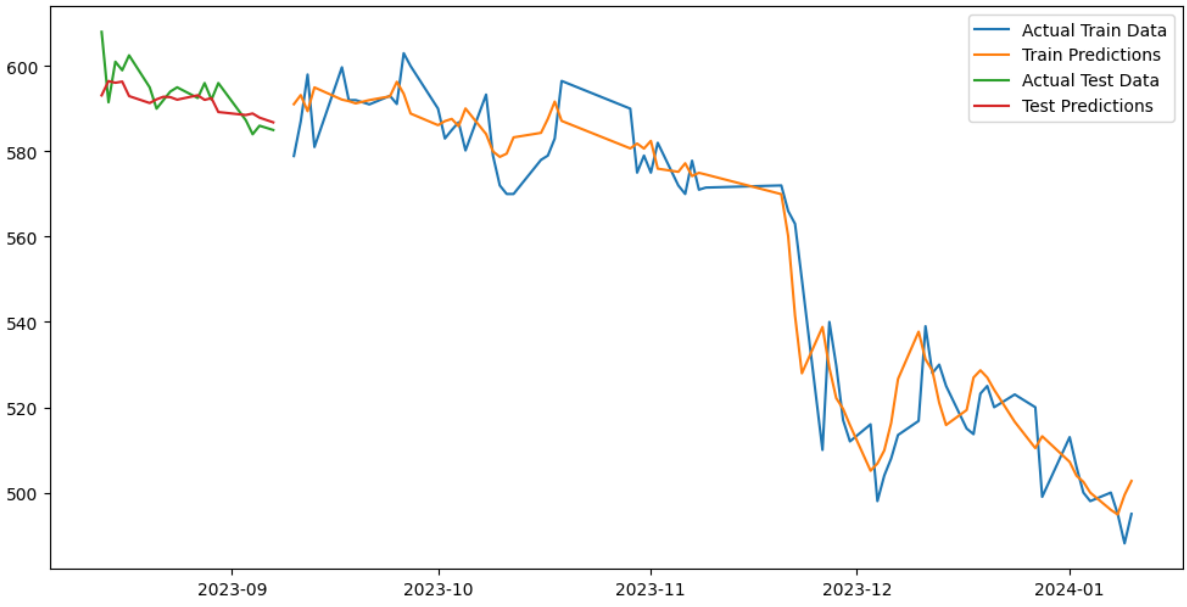

Figure 6.5: LSTM prediction

	SYMBOL	LTP	CHANGE	HIGH	LOW	OPEN	PREV. CLOSE
increase-row	ACLBSL	690.00	1.47	693.00	655.00	671.00	4,900
decrease-row	ADBL	236.00	-1.54	242.00	235.00	240.00	14,493
decrease-row	ADBLD83	1,020.00	-0.49	1,020.00	1,020.00	1,020.00	40
decrease-row	AHL	447.10	-1.93	455.90	446.00	447.00	1,926
decrease-row	AHPC	188.00	-0.79	192.90	186.10	192.90	71,383
decrease-row	AKJCL	228.70	-0.13	233.00	227.00	229.00	24,905
decrease-row	AKPL	188.20	-0.9	192.70	186.20	186.20	112,715
nochange-row	ALBSL	700.00	0	710.00	687.10	707.00	3,750
nochange-row	ALICL	600.00	0	609.00	590.00	601.00	37,531
increase-row	ANLB	2,238.00	9.22	2,253.90	2,079.00	2,079.00	8,512
decrease-row	API	191.10	-0.21	195.00	189.00	192.50	141,916

Figure 6.6: Web Application Home Page

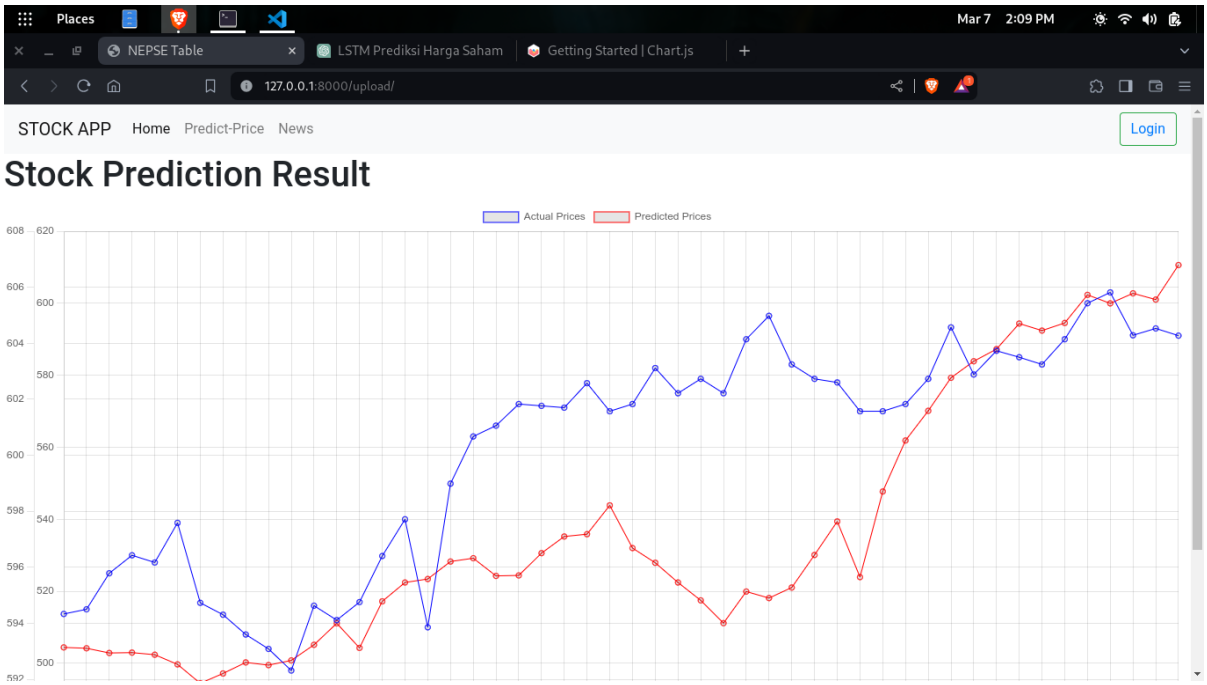

Figure 6.7: Web Application Data visualization Section

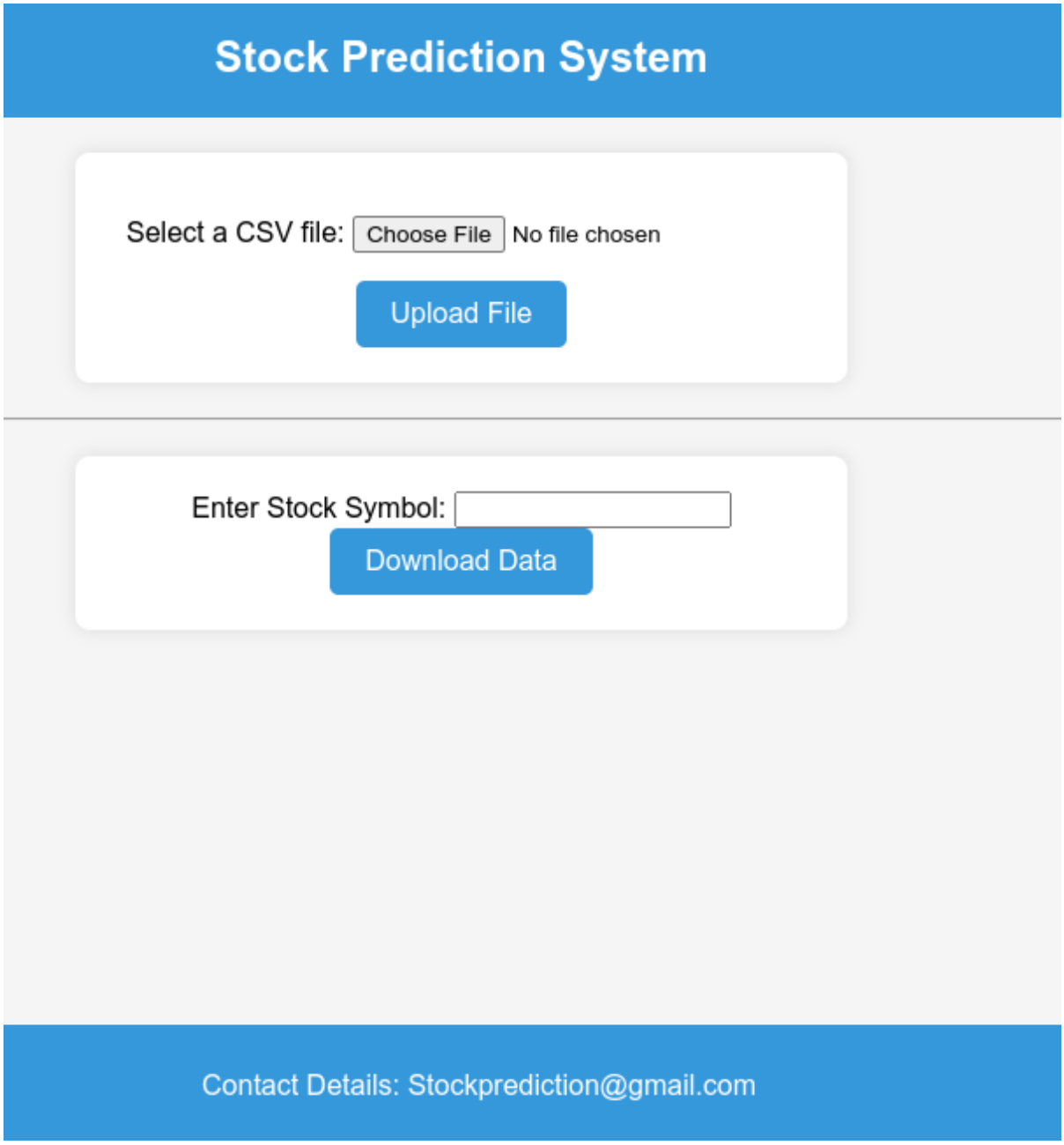

Figure 6.8: Web Application Form Section

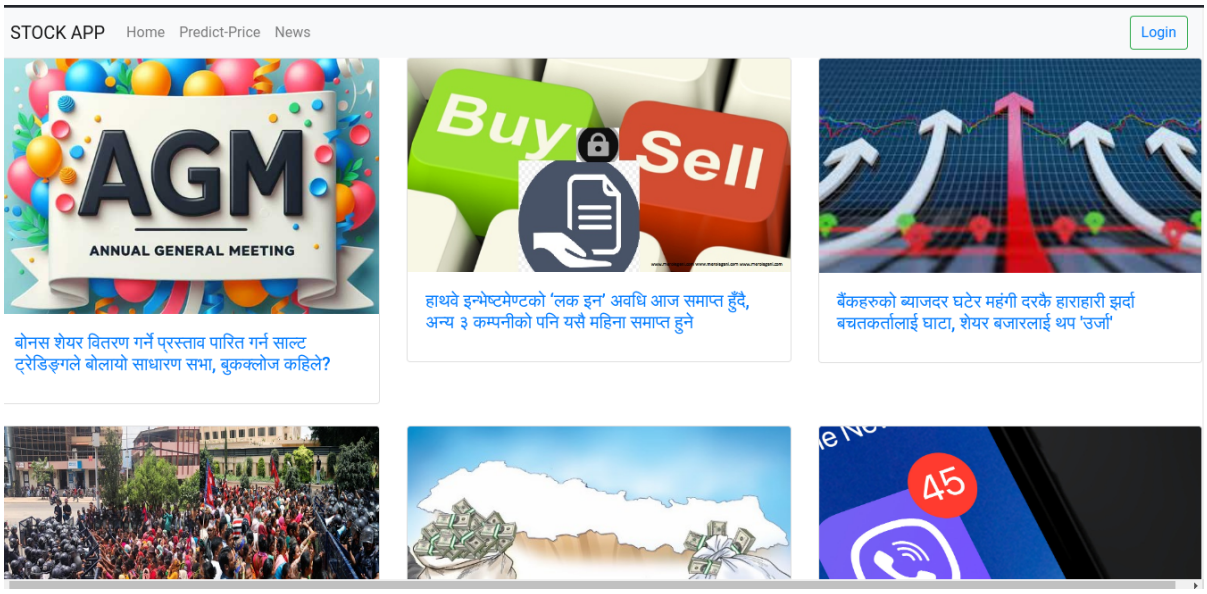

Figure 6.9: Web Application News Section