

PROJECT REPORT
ON
“STOCK PRICE PREDICTION WEB APP”
BY
“KRISHNA ASHOKRAO TIKULE”

MASTERS OF COMPUTER APPLICATION
SAVITRIBAI PHULE PUNE UNIVERSITY



ATSS's
INSTITUTE OF INDUSTRIAL AND COMPUTER MANAGEMENT
AND RESEARCH NIGDI
PUNE 411044
2024-2025

Index

Sr.no	Topic	Page No.
1	INTRODUCTION	
	1.1 Abstract	4
	1.2 Existing System	5
	1.3 Need for System	5
	1.4 Scope of System	6
	1.5 Operating Environment - Hardware and Software	6
	1.6 Brief Description of Technology Used	7
2	PROPOSED SYSTEM	
	2.1 Proposed System	12
	2.2 Objectives of System	13
	2.3 Features of Proposed System	13
	2.4 Users of System	14
3	ANALYSIS & DESIGN	
	3.1 System Requirements (Functional and Non-Functional requirements)	15
	3.2 Table Structure (in case back end is a database)	16
	3.3 Use Case Diagrams	17
	3.4 Class Diagram	18
	3.5 Activity Diagram	19
	3.6 Deployment Diagram	20
	3.7 Module Hierarchy Diagram	21
4	USER SCREENS & REPORTS	
	4.1 Input Screens	22
	4.2 Output Screens	24
	4.3 Reports	24
5	CODING	
	5.1 Algorithm	25
	5.2 Code Snippets	26
6	TESTING	
	6.1 Introduction	34
	6.2 Test Plan	34
	6.3 Test cases	35
7	RESULT & DISCUSSION (for AIML projects)	36
8	LIMITATION OF PROPOSED SYSTEM	38
9	PROPOSED ENHANCEMENT	39
10	CONCLUSION	40
11	BIBLIOGRAPHY	41
12	USER MANUAL (All screens with proper description/purpose Details about validations related to data to be entered.)	42

Acknowledgement

I would like to express my heartfelt gratitude to everyone who supported me in the development of this Stock Prediction Web App. This project would not have been possible without the guidance, mentorship, and resources provided by my institution, IICMR, and the invaluable support from my faculty and peers.

First and foremost, I am deeply grateful to my project advisor, for their continuous support, insightful feedback, and encouragement throughout this project. Their expertise in machine learning and web development was crucial to the completion of this app, and their guidance helped shape this project from inception to its final form.

I would also like to thank my friends and classmates for their collaboration and constructive feedback during the various stages of development. Their valuable suggestions improved both the design and functionality of the app.

Finally, I extend my gratitude to my family, who provided me with the moral and emotional support necessary to tackle the challenges of this project. Their unwavering belief in my potential motivated me to persevere and complete this endeavor.

Thank you all for your invaluable contributions and encouragement.

INTRODUCTION

The stock market has long been a cornerstone of investment strategy and financial growth for both individual investors and institutions. However, despite its potential for profit, the inherent volatility of the stock market poses significant challenges to investors. Stock prices are influenced by a vast array of variables, including economic indicators, global events, and company-specific factors. Traditional investment approaches have thus relied heavily on financial expertise and qualitative analysis to predict market behavior. With the rise of machine learning, however, the process of forecasting stock prices has been revolutionized, allowing for complex data patterns to be analyzed and interpreted in a way that was previously unattainable.

Our project, a Stock Prediction Web Application, utilizes an advanced Long Short-Term Memory (LSTM) neural network model to predict future stock price trends based on historical data. LSTM models, a type of recurrent neural network (RNN), are particularly well-suited for time-series forecasting as they can capture both short- and long-term dependencies in data sequences. This web application aims to serve as a powerful yet user-friendly tool for both novice investors and experienced financial analysts. It enables users to make data-driven predictions, visualize stock trends, and manage their favorite stocks within a personalized, interactive environment.

The application's main goal is to provide a seamless and efficient experience, integrating data retrieval, prediction, and visualization. Leveraging machine learning, this system provides users with predictive insights into stock price movements, making it an invaluable asset for personal or professional investment strategies. By presenting stock predictions alongside customizable data visualization options, the application aids users in making informed financial decisions and enhances their engagement with the stock market.

1.1 Abstract

In an increasingly data-driven world, accurate financial forecasting, particularly in stock market predictions, has become crucial for investors, analysts, and financial institutions. This project presents a comprehensive web application designed for predicting stock prices using machine learning techniques, specifically leveraging a Long Short-Term Memory (LSTM) neural network model. LSTM models are particularly suited to time-series analysis, making them ideal for predicting stock trends based on historical data patterns. The application is user-centric and provides a robust interface that allows authenticated users to access predictions, manage favorite stocks, and view real-time data visualizations, including historical price charts and moving averages.

This web application is developed using Streamlit as the front end, providing a seamless and interactive experience, while the back end integrates Python libraries for data processing and model deployment. The predictive model has been trained on historical stock data sourced from Yahoo Finance, which is then preprocessed and normalized using the MinMaxScaler to improve model accuracy. After preprocessing, the data is fed into the LSTM model, which

learns from past price movements to predict future trends. The app enables users to input stock ticker symbols to obtain predictions, view historical data, and access a range of visualizations to understand market patterns better.

The platform features essential functionalities such as user authentication and profile management, allowing users to register and securely log in to access advanced features. A unique aspect of this app is the favorite stock management system, which allows users to save specific stock tickers for easy access. These favorites are displayed on the profile page and can be removed as desired, ensuring that users have control over their personalized stock portfolio within the app. Additionally, the app's prediction page provides an interactive section where users can visualize the predicted trends over a selected period, along with historical trends, enhancing their decision-making with comprehensive data insights.

The scope of this project not only includes technical forecasting but also emphasizes usability and accessibility. Designed to operate on a standard internet-enabled device, the application ensures that users can access its features regardless of technical background. By integrating a machine learning model into an intuitive web interface, this project bridges the gap between advanced financial forecasting and user-friendly design, empowering users to make informed decisions in the volatile stock market environment.

This stock prediction web application represents a powerful tool for investors and financial enthusiasts, combining machine learning, data visualization, and user experience in a single platform. The application's approach highlights the efficacy of LSTM models in financial predictions and sets the foundation for further developments, such as adding more complex models or expanding data sources, to enhance prediction accuracy and system versatility.

1.2 Existing System and Need for System

Current stock prediction systems primarily fall into three categories:

Basic Financial Tools: These typically provide static data, such as historical stock prices and simple trend graphs, without predictive insights.

Advanced Analytical Platforms: Systems like Bloomberg Terminal or Eikon provide comprehensive data analytics and predictions but are highly complex and expensive, making them inaccessible for many users.

Machine Learning-Based Applications: Some applications offer predictive capabilities, but they often lack personalization, making them less user-centric. These platforms are also prone to issues with data visualization and fail to provide an intuitive user interface for less technical audiences.

The need for a streamlined, accessible, and effective prediction tool is thus evident. Our application addresses this gap by offering:

User-Friendly Accessibility: A simple login, profile management, and predictive interface, making it approachable for non-technical users.

Personalization Options: Features such as favorite stocks, allowing users to manage their portfolio within the application.

Advanced Prediction Model: The LSTM model, specifically trained for stock data, improves prediction reliability by capturing intricate patterns in time-series data.

These features combined make the application a useful tool that bridges the gap between technical market analysis platforms and overly simplified stock-tracking applications.

1.3 Scope of Work

The Stock Prediction Web Application was developed with a comprehensive set of goals, focusing on functionality, accessibility, and ease of use. The major features of the system include:

User Authentication and Profile Management: Users can register, log in, and view their profile. The profile page includes features for managing personal information and a favorites section for selected stocks.

Stock Prediction Engine: The core of the application is a machine learning model that predicts stock prices. Users can input specific stock tickers and receive a forecast based on the LSTM model's analysis of historical data.

Interactive Data Visualization: Users can view stock trends through graphs that display historical data, moving averages, and predicted versus actual prices.

Favorite Stocks Management: Users can add or remove stocks from their favorites, which are displayed on their profile page for quick access.

Responsive Interface: Built using Streamlit, the application interface is user-friendly, making it simple for users to navigate between pages, view data, and manage predictions.

This application thus caters to a diverse user base, including:

Individual Investors: Users with minimal financial knowledge but interested in stock investments.

Data Analysts and Financial Enthusiasts: Users with a background in data science or finance who wish to explore machine learning-driven financial predictions.

Educational Use: An excellent tool for academic settings, where students can learn about machine learning applications in finance.

1.4 Operating Environment

To ensure smooth deployment and usability, the following hardware and software specifications were considered:

Hardware Requirements:

Processor: A minimum Intel i5 processor or equivalent
RAM: At least 8GB for efficient data processing and model inference
Storage: Approximately 500 MB for the application files and additional space for dataset storage
Network: An internet connection for real-time data retrieval from the Yahoo Finance API

Software Requirements:

Programming Language: Python 3.7 or higher, which provides an extensive ecosystem for machine learning and data visualization
Framework: Streamlit, which allows rapid deployment of web applications with a responsive interface
Libraries: TensorFlow and Keras, Pandas and NumPy, scikit-learn, Matplotlib
Yahoo Finance API: For fetching live stock data, which is essential for real-time predictions and data updating
Database: SQLite is used to store user data and favorite stocks due to its simplicity and compatibility with Python-based applications

1.5 Detail Description of Technology Used

The Stock Prediction Web Application leverages several key technologies, each playing a unique role in creating a robust, user-friendly, and effective prediction tool. This section provides a comprehensive overview of the technologies used, including their functionalities, advantages, and specific applications in our project.

1. Python

Python serves as the primary programming language for this project due to its simplicity, extensive libraries, and versatility in handling data processing and machine learning tasks. Python's syntax is user-friendly, making code readability and maintenance more efficient. Additionally, Python supports a vast range of packages specifically designed for data science, machine learning, and web application development, making it ideal for an end-to-end data-driven application like this one.

Key Packages Utilized:

Pandas: This library is essential for data manipulation and analysis. It allows us to load, clean, process, and transform stock data efficiently, making it ready for model training and prediction.

NumPy: Used for numerical operations, NumPy supports efficient matrix computations and data transformation, which are critical for preparing the data in a format suitable for time-series analysis.

Matplotlib: An essential library for data visualization, Matplotlib allows us to create interactive line charts, histograms, and other visual representations, which are used to plot stock prices, trends, and predictions for users.

2. Streamlit

Streamlit is a powerful open-source framework designed specifically for creating data-driven web applications. It allows developers to build intuitive, interactive, and real-time web interfaces with minimal code, as Streamlit abstracts much of the complexity involved in traditional web development.

User Interface: Streamlit's ability to create sliders, buttons, drop-downs, and text inputs makes it possible to create a dynamic and user-friendly interface that responds instantly to user actions.

Real-Time Interaction: Streamlit supports real-time data updates, enabling users to input stock tickers and view instant predictions without having to refresh the page.

Ease of Deployment: Streamlit apps can be deployed on cloud platforms like Streamlit Share or other hosting environments, making it easy to access and maintain. Its lightweight nature also makes the app responsive across devices, enhancing the user experience.

3. TensorFlow and Keras

TensorFlow, developed by Google, is one of the most widely used machine learning frameworks, and it provides a range of tools for deep learning, data processing, and model evaluation. Keras, a high-level API within TensorFlow, simplifies the process of creating, training, and deploying deep learning models, making it ideal for our stock prediction model, which relies on sequential data processing.

Long Short-Term Memory (LSTM) Network: An LSTM is a type of Recurrent Neural Network (RNN) specifically designed for handling sequential data. It's known for its ability to retain information over time and capture temporal dependencies in time-series data, which is essential in stock price prediction. In our application, the LSTM model is trained on historical stock prices, learning to recognize patterns and make informed predictions about future prices.

Scalability and Model Optimization: TensorFlow's efficient execution model enables us to train the LSTM model on large datasets, and its optimizers allow for fine-tuning parameters, improving the accuracy and performance of our predictions.

Model Deployment: Using TensorFlow with Keras allows us to deploy the model seamlessly within the Streamlit app, ensuring that predictions are quick and responsive.

4. Yahoo Finance API

The Yahoo Finance API provides access to a vast repository of financial data, including real-time and historical stock prices, trading volume, and financial news. Integrating this API ensures that our predictions and analyses are based on up-to-date market data, enhancing the application's reliability and relevance.

Real-Time Data Retrieval: Users can input a stock ticker symbol, and the application retrieves real-time data for that specific stock. This feature allows users to make predictions based on the most current market conditions.

Historical Data: The API also provides historical data, which is crucial for training the LSTM model. We retrieve and preprocess this historical data to identify trends and patterns over time, training the model on a representative sample of data to improve its forecasting capabilities.

5. SQLite

SQLite is a lightweight and self-contained relational database management system that stores user information, login credentials, and favorite stocks. This database is highly compatible with Python and requires minimal configuration, making it well-suited for our project, which does not require complex data management.

User Management: SQLite securely stores user credentials, enabling the authentication and authorization functionality within the app. This feature is crucial for managing user access and personalizing the app experience for each user.

Favorite Stocks Storage: Users can add or remove favorite stocks, which are stored in the SQLite database. This personalization ensures that users can easily access stocks they follow, enhancing engagement and user experience.

Efficient Data Access: SQLite's design allows fast data access and query execution, making it an excellent choice for real-time applications where speed is a priority.

6. Scikit-Learn

Scikit-Learn is a comprehensive machine-learning library in Python that offers tools for data preprocessing, model selection, and evaluation. It complements TensorFlow by providing essential data transformation capabilities.

Data Preprocessing: Scikit-Learn's tools for normalization and scaling are used to prepare the stock price data before it's fed into the LSTM model. Standardizing data is essential in time-series forecasting as it helps the model understand patterns more effectively.

Evaluation Metrics: Using Scikit-Learn, we calculate metrics such as Mean Absolute Error (MAE) and Mean Squared Error (MSE) to evaluate the model's performance and tune parameters as necessary to achieve optimal results.

7. Matplotlib

Matplotlib is the primary library used for data visualization in this project. It allows us to create visually appealing and informative charts, which enhance the user's ability to interpret the predictions generated by the application.

Prediction Graphs: Matplotlib enables the plotting of prediction results in relation to historical data, giving users a visual understanding of how the predicted prices compare to past trends.

Moving Averages: The application also includes moving average lines on stock charts, which provide additional context to users by smoothing out fluctuations in stock prices and revealing longer-term trends.

8. Pandas and NumPy

Pandas and NumPy are the backbone libraries for data manipulation and numerical operations in this project. They provide tools for handling, processing, and preparing the data used in predictions.

Data Cleaning: Pandas is used to filter and clean the raw data retrieved from the Yahoo Finance API, removing any missing or irrelevant information before analysis.

Data Transformation: Using NumPy, we transform data into arrays suitable for time-series analysis, structuring it in a format that the LSTM model can process efficiently.

Data Analysis: Pandas and NumPy enable detailed exploration of stock data, allowing us to calculate metrics like moving averages and price volatility, which are crucial for meaningful data visualization.

9. Application Integration and Workflow

Each of these technologies plays a specific role in the development and deployment of the Stock Prediction Web Application:

Front-End Interface (Streamlit): Provides a simple, interactive interface for users.

Back-End Processing (TensorFlow, Scikit-Learn): Handles data preprocessing, model training, and prediction.

Data Handling (Pandas, NumPy): Manages data cleaning, transformation, and manipulation.

Database (SQLite): Ensures secure and efficient user data management.

API Integration (Yahoo Finance): Facilitates real-time data retrieval.

The synergy of these technologies ensures that the application is user-friendly, accurate, and responsive, providing users with a powerful tool for stock price analysis and prediction. This technological ecosystem, centered around Python and supported by dedicated machine learning, data manipulation, and visualization libraries, makes the application robust and scalable, allowing it to deliver real-time, data-driven insights to users in an intuitive and accessible manner.

PROPOSED SYSTEM

The proposed Stock Prediction Web Application aims to address the limitations of traditional stock prediction methods by using a machine learning model and providing users with a real-time, interactive interface for stock analysis. This system will empower users to make data-driven investment decisions by enabling them to access predictions on stock prices based on historical data, identify trends, and manage their favorite stocks. Below is a detailed outline of the proposed system and its core components.

2.1 Proposed System

The proposed system is a comprehensive web application with the following key components:

User Registration and Authentication:

The app includes a login and registration functionality to ensure secure and personalized access. User profiles are created and stored in a database, enabling each user to view their preferred settings, saved stocks, and historical interactions.

Stock Prediction Module:

A machine learning-based prediction module, powered by a Long Short-Term Memory (LSTM) neural network, generates stock price forecasts based on historical stock data. Users can input any valid stock ticker, and the model predicts future trends for the given stock. The model is trained on past stock data, considering trends, seasonality, and other factors, and delivers predictions that reflect the likely future movements of the stock price.

Favorite Stocks Management:

Users can save specific stocks to a "Favorites" list, making it easier to track frequently viewed stocks. The user profile page will display these favorites, allowing users to add or remove stocks as desired, with the app's database managing these entries.

Data Visualization and Analysis:

The system provides visual insights into stock performance, displaying historical prices, predicted trends, and additional information like moving averages. This information helps users better interpret the model's predictions and make informed investment decisions. Interactive charts and trend graphs show both historical data and forecasted values, allowing users to visually track stock price movements over time.

Real-Time Data Retrieval and Updates:

Using the Yahoo Finance API, the application retrieves real-time stock data, ensuring that predictions are based on the latest available information. Users are provided with current stock prices and relevant market information, adding further context to the predictions.

2.2 Objectives of System

The objectives of the Stock Prediction Web Application are:

To Provide Reliable Predictions:

Leverage machine learning models to deliver accurate stock price forecasts based on historical data, helping users make informed decisions.

To Enhance User Accessibility:

Create an interactive and intuitive user interface for easy navigation, enabling users of all backgrounds to interact with the app seamlessly and obtain valuable insights.

To Offer Real-Time Stock Tracking and Alerts:

Allow users to track their favorite stocks in real-time, access the latest data, and save stock predictions for easy future access.

To User-Specific Customization:

Customize the user experience by enabling users to manage their profiles and favorite stocks, creating a more personalized experience within the app.

To Scalability and Flexibility:

Design the system to handle various stock tickers, accommodate more users over time, and be adaptable for the integration of additional features such as sentiment analysis and advanced portfolio tracking.

2.3 Features of the Proposed System

The proposed stock prediction web application is designed to provide a seamless and feature-rich experience for users, enabling them to access, analyze, and predict stock prices effectively. Below are the key features of the system:

User Authentication:

Secure login and registration system to ensure user data privacy.
Session management to maintain user authentication throughout usage.

Stock Prediction:

Predict future stock prices using machine learning algorithms, providing insights for decision-making. Visualization of prediction results through interactive graphs and charts for better comprehension.

User Profile Management:

Each user has a personalized profile where they can manage their preferences.
Option to add and remove favorite stocks for quick access.

Historical Stock Data Visualization:

Display historical stock price trends through line charts and candlestick charts.
Analysis of past trends to aid in understanding market movements.

News Integration:

A dedicated news page that fetches and displays the latest stock-related news from Alpha Vantage. Option to view all news or filter news for a specific stock ticker, keeping users informed about market events.

Interactive User Interface:

User-friendly interface built using Streamlit for easy navigation.

Responsive design ensures accessibility on both desktops and mobile devices.

Favorite Stocks Management:

Users can add stocks to their favorites list and manage them through their profile page.

Remove unwanted stocks from the favorites list with a simple button click.

2.4 Users of the System

The Stock Prediction Web Application is intended for a wide range of users who are interested in stock markets and financial analysis. Key user categories include:

Individual Investors, Individuals investing in stock markets can use the app to analyze trends, track stocks of interest, and make data-informed decisions regarding their portfolios.

Financial Enthusiasts and Analysts:

Users with a keen interest in financial markets and technical analysis can leverage the prediction tool to explore price trends and validate their own investment hypotheses.

Beginners in Stock Markets:

Novice investors who are just starting in stock markets can benefit from the application's straightforward, easy-to-use interface, gaining insights that help them make informed decisions while learning more about market dynamics.

Students and Researchers:

Students and researchers in fields such as finance, machine learning, and data science can use the app as a practical tool to understand time-series prediction models and explore the real-world applications of LSTM networks.

Summary of Proposed System

The proposed Stock Prediction Web Application is a dynamic, user-friendly tool designed to provide personalized stock price predictions and interactive market insights. By incorporating machine learning, real-time data access, and user-focused features, the system aims to support a wide range of users in making informed decisions, thus enhancing their understanding of stock trends and investment strategies. Through secure user management, predictive analytics, and intuitive visualization, this application will serve as a valuable resource for anyone interested in the stock market.

ANALYSIS & DESIGN

3.1 System Requirements

Functional Requirements

The functional requirements specify the core functionalities and tasks the system must perform to meet user needs effectively.

User Authentication

- Users must be able to register with a unique username and password.
- Users must be able to log in securely to access personalized features.
- Users must remain logged in during their session, with an option to log out.

Stock Prediction

- The system must accept stock ticker input from the user to generate predictions.
- Predictions must be displayed as visual charts, including line graphs.
- The system must provide an option to add the stock to the user's favorites.

Favorite Stocks Management

- Users must be able to view, add, and remove stocks from their favorites list.
- The favorite stocks list must persist in the database and reflect changes dynamically.

Stock News Integration

- Users must be able to view the latest stock-related news.
- The news page must allow users to filter by a specific stock ticker or view general stock market news.

Profile Management

- Users must be able to view their personal information and favorite stocks on the profile page.

Database Management

- The app must securely store user data, such as login credentials and favorite stocks, in a database.

Non-Functional Requirements

- **Performance:** The app should display predictions and news within a reasonable time.
- **Usability:** The app should have a simple and intuitive interface, making it easy for users to navigate.
- **Reliability:** The app should provide accurate predictions using the trained machine learning model.
- **Security:** User data, especially passwords, must be stored securely and protected from unauthorized access.
- **Compatibility:** The app should work seamlessly on common web browsers, such as Chrome and Firefox.
- **Maintainability:** The codebase should be organized to allow for easy updates and debugging.

3.2 Table specifications

Database users.db

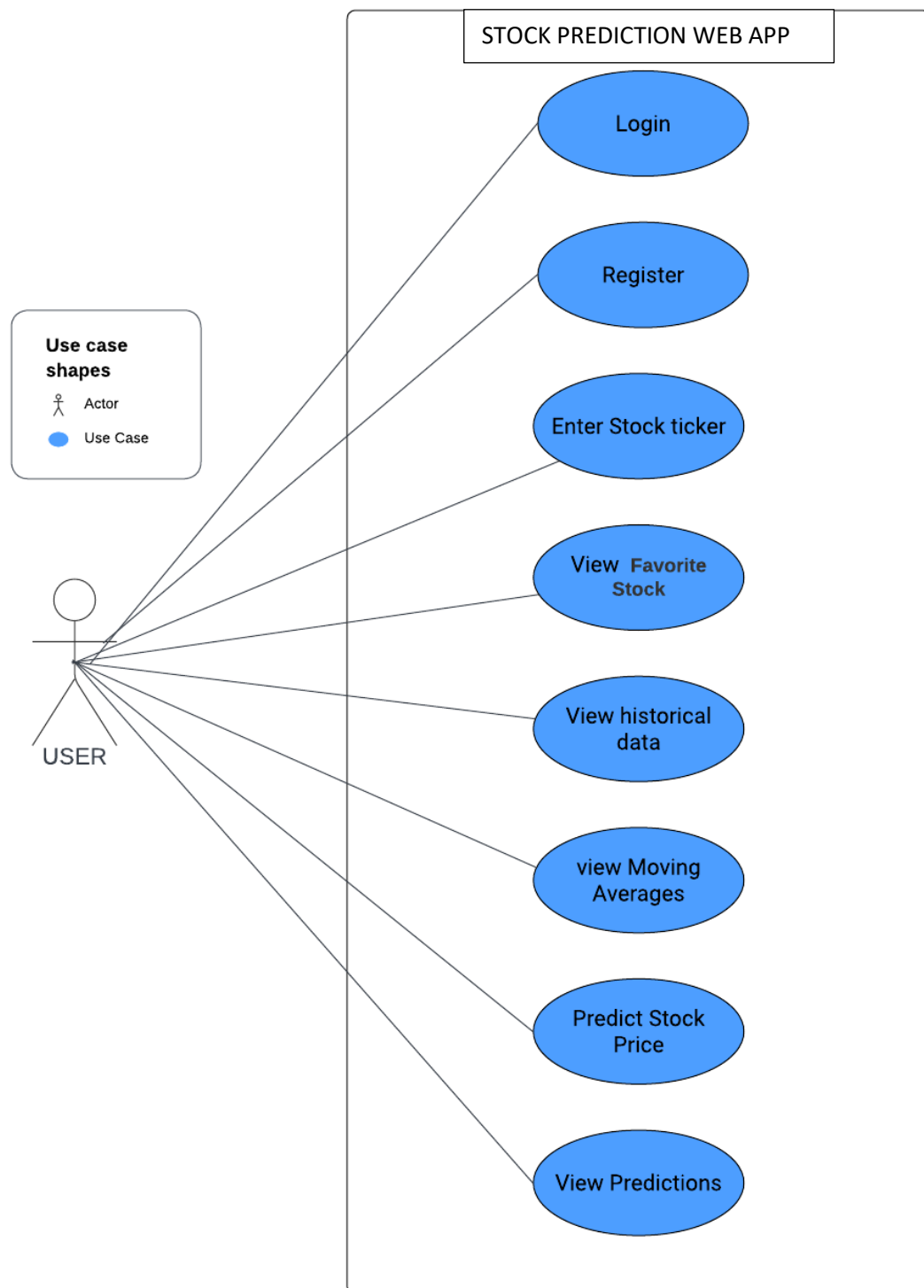
Table structure for table users

Column	Type	Null	Constraint
username	Text	No	Primary key
password	Text	No	
email	Text	No	
full_name	Text	No	

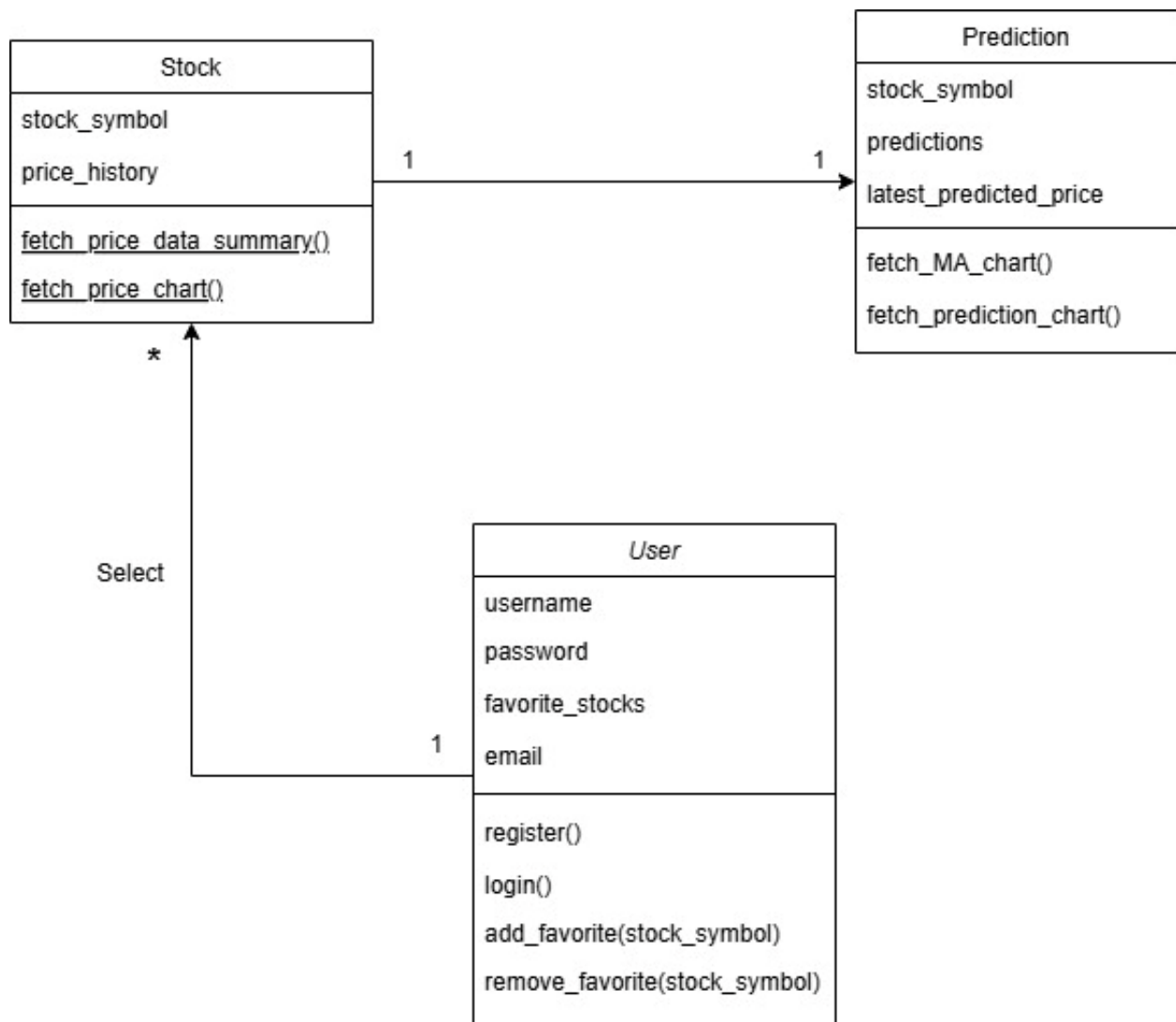
Table structure for table favorite_stocks

Column	Type	Null	Constraint
<i>Username</i>	Text	No	Foreign key
Stock_sumbol	Text	No	

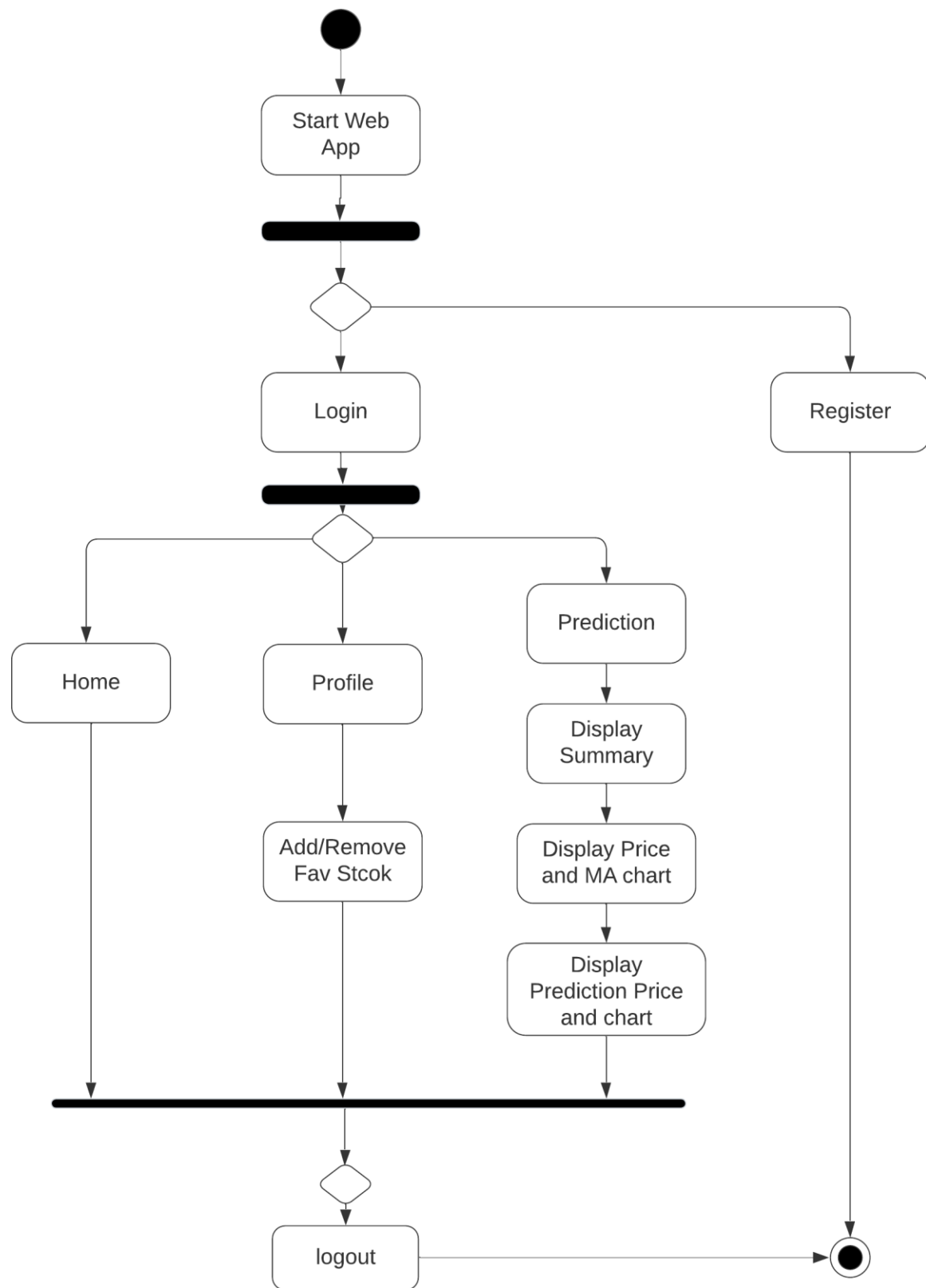
3.3 Use Case Diagrams



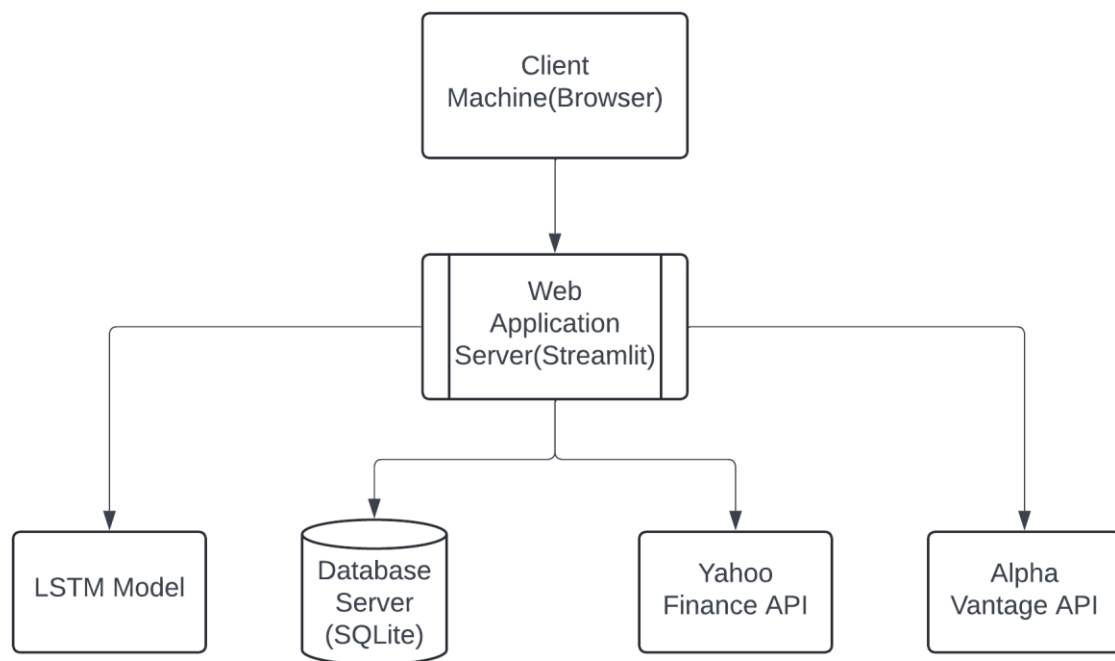
3.4 Class Diagram



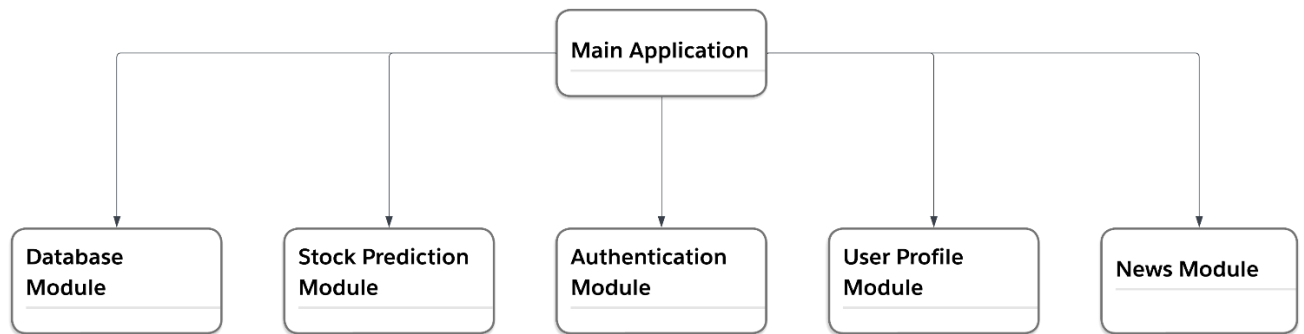
3.5 Activity diagram



3.6 Deployment Diagram



3.7 Module Hierarchy Diagram



USER SCREENS & REPORTS

4.1 User Interface Input Screens

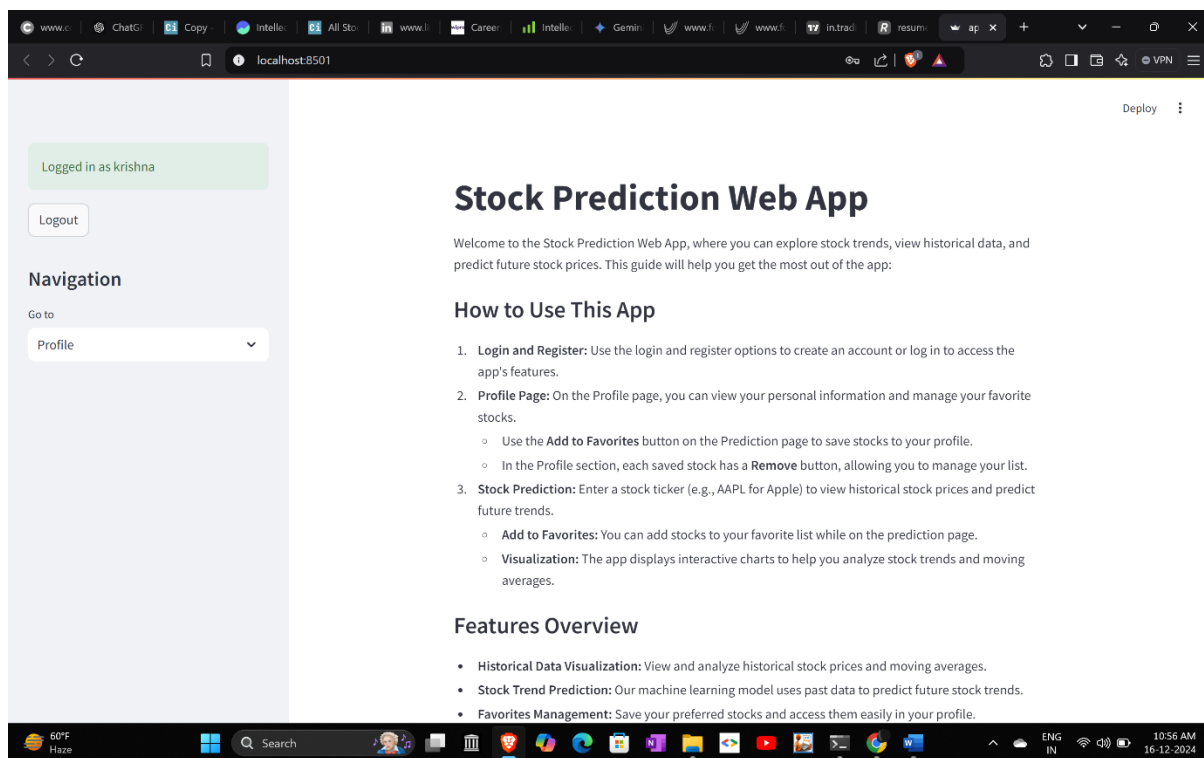
1.User Login:

The screenshot shows a web browser window at localhost:8501. On the left is a navigation sidebar with a 'Go to' dropdown menu set to 'Home'. The main content area has a 'Login/Register' dropdown menu set to 'Login'. Below this is the 'Login' section with input fields for 'Username' (containing 'krishna') and 'Password' (masked with dots). A 'Login' button is positioned below the password field. A red border highlights the password field, with a hint 'Press Enter to apply' and an eye icon. Below the login form is a 'Welcome to Stock Prediction Web App' message and a note: 'Please log in or register to access features.' The browser's taskbar at the bottom shows the time as 10:55 AM on 16-12-2024.

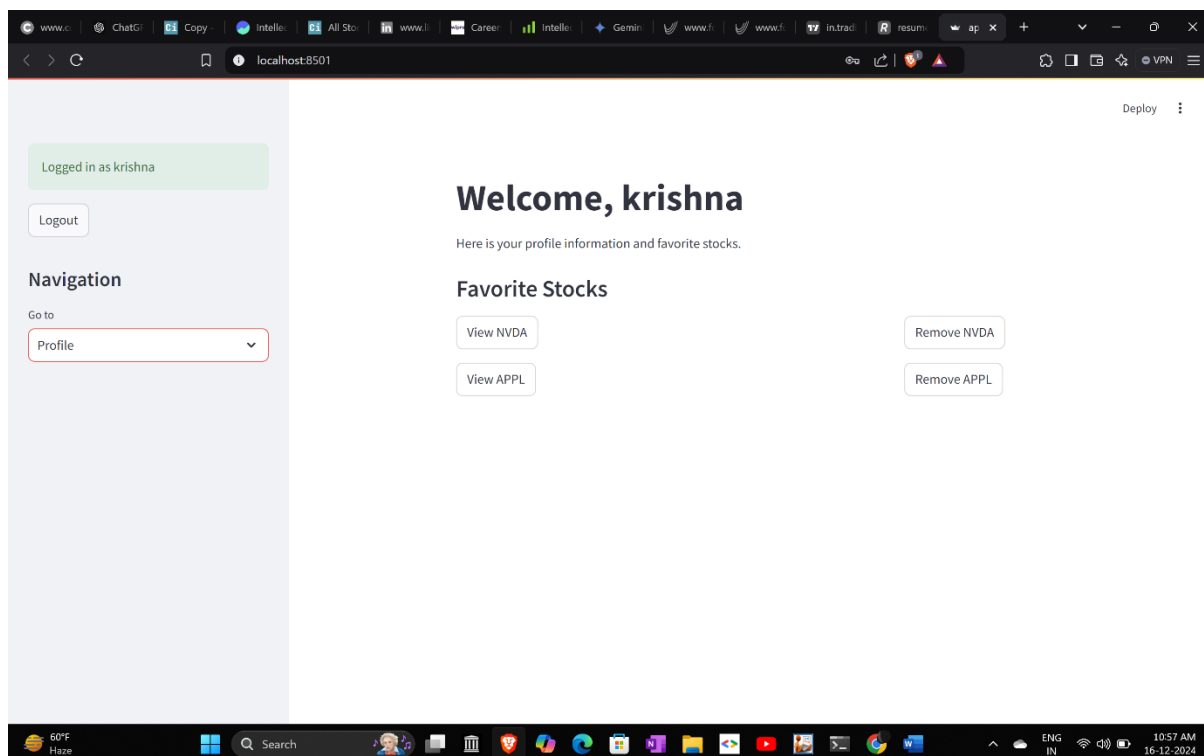
2.User Registration:

The screenshot shows the same web browser window at localhost:8501, but the 'Login/Register' dropdown menu is now set to 'Register'. The 'Register' section contains input fields for 'Full Name' (containing 'Krishna Tikule'), 'Email' (containing 'tikulekrishna@gmail.com'), 'Username' (containing 'krishna'), and 'Password' (masked with dots). A 'Register' button is located below the password field. A red border highlights the email field, with a hint 'Press Enter to apply' and an eye icon. Below the registration form is the same 'Welcome to Stock Prediction Web App' message. The browser's taskbar at the bottom shows the time as 10:56 AM on 16-12-2024.

3. Home:

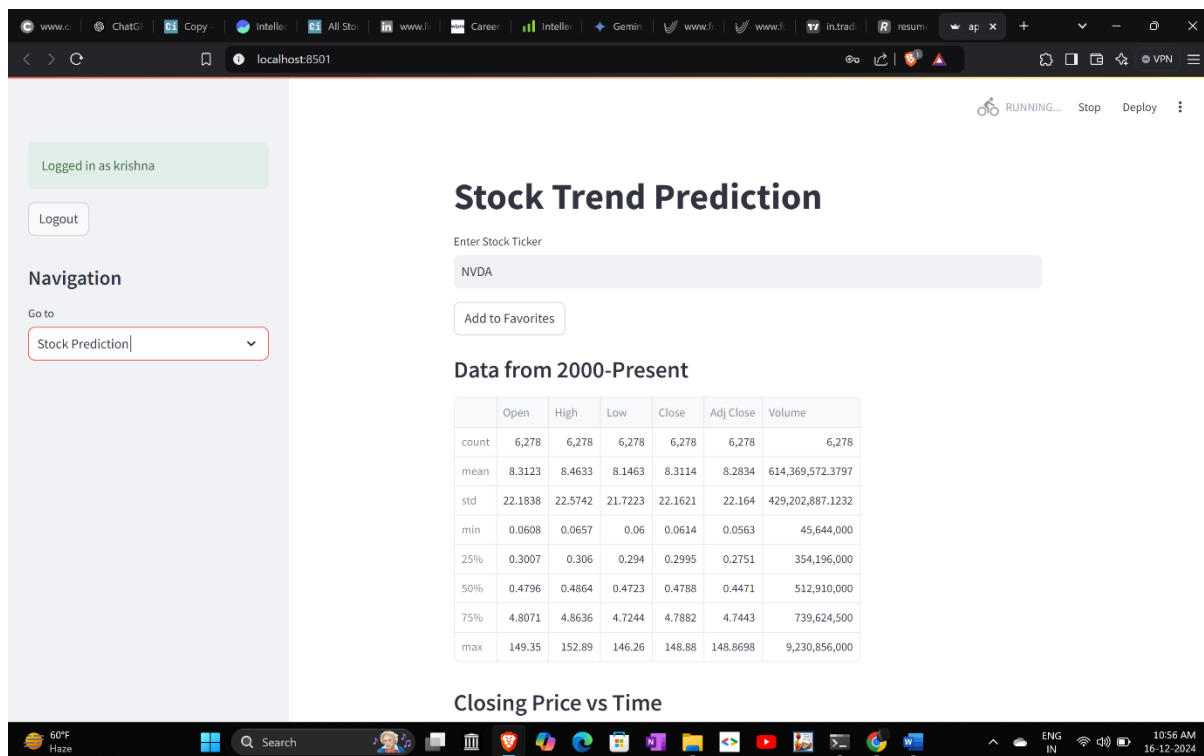


4. Profile Page:

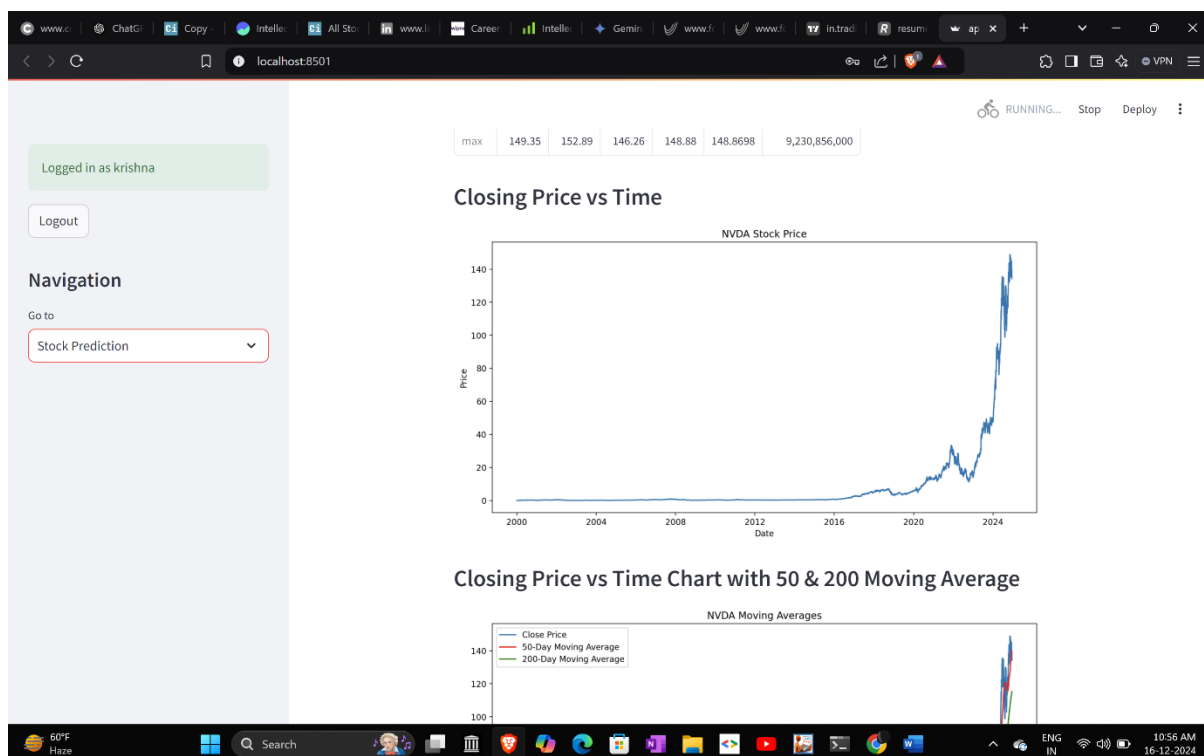


4.2 Output Screens

1. Trend Prediction:



4.3 Reports



Logged in as krishna

Logout

Navigation

Go to

News

Latest Stock News

Enter a stock ticker for specific news

NVDA

Latest Stock News

News for NVDA

3 Market-Beating Tech Stocks to Supercharge Your Portfolio in 2025 and Beyond

20241215T154500

The technology sector was a party in 2024. Thanks to their robust fundamentals and red-hot growth opportunities, these three prominent tech stocks are poised to continue the fun.

[Read more](#)

The Best Stocks to Invest \$50,000 in Right Now

60°F Haze

Search

10:57 AM 16-12-2024

Logged in as krishna

Logout

Navigation

Go to

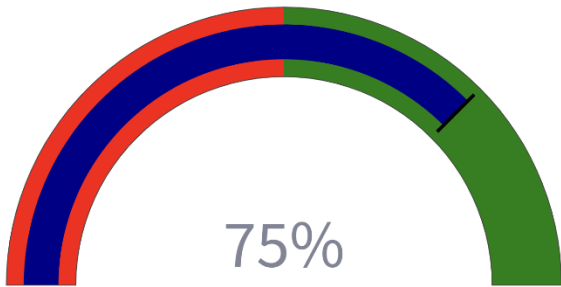
Stock Prediction

Time

Latest Predicted Price for NVDA

The predicted stock price for the latest day is: \$122.11

Stock Sentiment Indicator



75%

60°F Haze

Search

10:56 AM 16-12-2024

CODING

5.1 Algorithms

The coding of the stock prediction app follows a structured algorithm for each key functionality:

1. User Authentication:

- Input: Username and password from the user.
- Process: Check if the user is registering or logging in.
- Validate credentials against the database.
- Output: Log in the user if credentials match or show an error message.

2. Stock Prediction:

- Input: Stock ticker symbol from the user.
- Process: Fetch historical stock data using an API (e.g., Alpha Vantage).
- Preprocess the data and input it into the trained LSTM model.
- Generate future stock price predictions.
- Output: Display a graph of the predicted stock prices.

3. Favorite Stocks Management:

- Input: Stock ticker selected by the user.
- Process: Add the stock ticker to the user's favorite list in the database.
- Allow removal of stocks from the favorite list.
- Output: Update the favorites displayed on the profile page.

4. Stock News Integration:

- Input: Stock ticker or general news request.
- Process: Fetch relevant news articles using the Alpha Vantage News API.
- Filter news based on the selected stock ticker if applicable.
- Output: Display the latest news in a structured format.

App.py:

```

import streamlit as st
from login import auth, logout
from prediction import predict
from profile import user_profile
from database import connect_db

# Initialize database connection
connect_db()

# Authentication and navigation
auth()

# Sidebar navigation
st.sidebar.title("Navigation")
page = st.sidebar.selectbox("Go to", ["Home", "Profile", "Stock Prediction"])

# Ensure the user is logged in before showing the pages
if st.session_state['logged_in']:

    if page == "Home":
        st.title("Stock Prediction Web App")
        st.write("""
        Welcome to the Stock Prediction Web App, where you can explore stock trends, view
        historical data, and predict future stock prices.

        This guide will help you get the most out of the app:
        """)

        # App usage guide
        st.subheader("How to Use This App")
        st.write("""
        1. **Login and Register:** Use the login and register options to create an account or log
        in to access the app's features.

```

2. **Profile Page:** On the Profile page, you can view your personal information and manage your favorite stocks.

- Use the **Add to Favorites** button on the Prediction page to save stocks to your profile.

- In the Profile section, each saved stock has a **Remove** button, allowing you to manage your list.

3. **Stock Prediction:** Enter a stock ticker (e.g., AAPL for Apple) to view historical stock prices and predict future trends.

- **Add to Favorites:** You can add stocks to your favorite list while on the prediction page.

- **Visualization:** The app displays interactive charts to help you analyze stock trends and moving averages.

""")

Additional Information

st.subheader("Features Overview")

st.write("""

- **Historical Data Visualization:** View and analyze historical stock prices and moving averages.

- **Stock Trend Prediction:** Our machine learning model uses past data to predict future stock trends.

- **Favorites Management:** Save your preferred stocks and access them easily in your profile.

""")

Tips section

st.subheader("Tips for Accurate Predictions")

st.write("""

- Always verify the stock ticker before inputting it for prediction.
- Predictions are based on historical data and moving averages; consider these as trends, not guarantees.

- Use the **Add to Favorites** option to easily access and manage frequently tracked stocks.

""")

elif page == "Profile":

```

        user_profile() # Show user profile and favorite stocks

    elif page == "Stock Prediction":
        predict() # Prediction logic

    else:
        st.title("Welcome to Stock Prediction Web App")
        st.write("Please log in or register to access features.")

```

login.py:

```

import streamlit as st
from database import add_user, get_user

def login():
    st.title('Login')

    username = st.text_input('Username')
    password = st.text_input('Password', type='password')
    if st.button('Login'):
        user = get_user(username, password)
        if user:
            st.session_state['logged_in'] = True
            st.session_state['username'] = username
            st.success('Login successful')
            st.rerun()
        else:
            st.error('Invalid username or password')

def register():
    st.title('Register')

```

```

full_name = st.text_input('Full Name')
email = st.text_input('Email')
username = st.text_input('Username')
password = st.text_input('Password', type='password')

```

```

if st.button('Register'):
    add_user(username, password, full_name, email)
    st.success('Registration successful. Please log in.')

```

```

def logout():
    st.session_state['logged_in'] = False
    st.session_state['username'] = None
    st.success('Logged out successfully.')
    st.rerun()

```

```

def auth():
    if 'logged_in' not in st.session_state:
        st.session_state['logged_in'] = False

    if not st.session_state['logged_in']:
        login_option = st.selectbox('Login/Register', ['Login', 'Register'])
        if login_option == 'Login':
            login()
        else:
            register()
    else:
        st.sidebar.success(f"Logged in as {st.session_state['username']}")
        if st.sidebar.button('Logout'):
            logout()

```

prediction.py:

```

import streamlit as st
import yfinance as yf
from tensorflow.keras.models import load_model
import matplotlib.pyplot as plt
from database import add_favorite_stock
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler

def predict():
    st.title('Stock Trend Prediction')

    if 'username' not in st.session_state:
        st.warning('Please log in to use the prediction feature.')
        return

    user_input = st.text_input('Enter Stock Ticker', 'AAPL')
    favorite = st.button('Add to Favorites')

    if favorite:
        add_favorite_stock(st.session_state['username'], user_input)
        st.success(f'{user_input} added to favorites')

    df = yf.download(user_input, '2000-01-01')

    st.subheader('Data from 2000-Present')
    st.write(df.describe())

    # Closing price vs Time Chart
    st.subheader("Closing Price vs Time")
    fig, ax = plt.subplots(figsize=(12, 6))
    ax.plot(df['Close'], label='Close Price')

```

```

ax.set_xlabel('Date')
ax.set_ylabel('Price')
ax.set_title(f'{user_input} Stock Price')
st.pyplot(fig)

# Prediction logic here
# Closing price vs Time Chart with 50 & 200 Moving Average
st.subheader("Closing Price vs Time Chart with 50 & 200 Moving Average")
ma50 = df['Close'].rolling(50).mean()
ma200 = df['Close'].rolling(200).mean()
fig = plt.figure(figsize=(12, 6))
plt.plot(df['Close'], label='Close Price')
plt.plot(ma50, 'r', label='50-Day Moving Average')
plt.plot(ma200, 'g', label='200-Day Moving Average')
plt.title(f'{user_input} Moving Averages')
plt.xlabel('Date')
plt.ylabel('Price')
plt.legend()
st.pyplot(fig)

# Splitting data into training and testing
data_training = pd.DataFrame(df['Adj Close'][0:int(len(df) * 0.70)])
data_testing = pd.DataFrame(df['Adj Close'][int(len(df) * 0.70):])

scaler = MinMaxScaler(feature_range=(0, 1))
data_training_array = scaler.fit_transform(data_training)

model = load_model('keras_model.h5')

past_100_days = data_training.tail(100)
final_df = pd.concat([past_100_days, data_testing], ignore_index=True)
input_data = scaler.fit_transform(final_df)

```



```

x_test = []
y_test = []

for i in range(100, input_data.shape[0]):
    x_test.append(input_data[i-100:i])
    y_test.append(input_data[i, 0])

x_test, y_test = np.array(x_test), np.array(y_test)

y_predicted = model.predict(x_test)
scaler = scaler.scale_
scale_factor = 1 / scaler[0]

y_predicted = y_predicted * scale_factor
y_test = y_test * scale_factor

# Plot predictions vs. originals
st.subheader('Predictions vs Originals')
fig = plt.figure(figsize=(12, 6))
plt.plot(y_test, 'b', label='Original')
plt.plot(y_predicted, 'r', label='Predicted')
plt.title('Stock Price Prediction')
plt.xlabel('Time')
plt.ylabel('Price')
plt.legend()
st.pyplot(fig)

# Extract the most recent prediction (latest day predicted price)
latest_predicted_price = y_predicted[-1].item()

# Display the most recent predicted price

```

TESTING

6.1 Introduction

Testing ensures the functionality, reliability, and performance of the stock prediction web app. The process involves identifying bugs, verifying feature implementations, and confirming that the system meets user requirements. Testing is categorized into functional testing, non-functional testing, and user acceptance testing (UAT).

The app was tested in various scenarios to ensure that user authentication, stock prediction, favorite stock management, and news integration function as intended.

6.2 Test Plan

Objective:

The primary objective of the test plan is to verify that the stock prediction app works as expected, satisfies user requirements, and performs well under various conditions.

Scope:

The test plan covers the following functionalities:

- User authentication
- Stock price prediction
- Profile management
- News display

6.3 Test Cases

Test Case ID	Description	Input	Expected Output	Result
TC001	User Registration	Valid username, password, email	Account created successfully	Passed
TC002	User Login	Valid username and password	User logged in and redirected to the home page	Passed
TC003	User Login	Invalid credentials	Error message displayed	Passed
TC004	Stock Prediction	Valid stock ticker	Predicted prices displayed as a chart	Passed
TC005	Stock Prediction	Invalid stock ticker	Error message displayed	Passed
TC006	Add to Favorites	Stock ticker	Stock added to user's favorite list	Passed

Test Case ID	Description	Input	Expected Output	Result
TC007	Remove from Favorites	Stock ticker	Stock removed from user's favorite list	Passed
TC008	News Integration (All Stocks)	None	Latest stock-related news displayed	Passed
TC009	News Integration (Specific Stock)	Stock ticker	News related to the specific stock displayed	Passed
TC010	Sidebar Navigation	Click on "Profile"	Profile page displayed	Passed
TC011	Sidebar Navigation	Click on "Stock Prediction"	Stock prediction page displayed	Passed
TC012	Sidebar Navigation	Click on "News"	News page displayed	Passed

RESULT & DISCUSSION

7.1 Result

The Stock Prediction Web App successfully implements machine learning techniques to predict stock prices and provides an interactive user interface for users to manage their portfolios. Below are the key results achieved by the application:

Stock Price Prediction: Users can input a stock ticker, and the app generates stock price predictions using the trained LSTM model. Predictions are displayed in a clear, interactive graph format, enabling users to visualize trends effectively.

User Management: The app allows users to register, log in, and manage their profiles securely. Users can add and remove favorite stocks for personalized tracking.

News Integration: A dedicated news page provides users with the latest stock-related news, enhancing decision-making by keeping users informed.

User Experience: The app's design ensures a seamless and user-friendly experience, with intuitive navigation between pages. Features such as stock predictions, news, and favorites are easy to access and interact with.

7.2 Discussion

The project demonstrates the successful application of machine learning in financial forecasting and highlights the potential of AIML technologies in decision support systems. Below are detailed insights and observations:

1. Model Performance:

The LSTM model used for stock price prediction performed well in capturing temporal dependencies within stock data. While the model's accuracy varied depending on the stock and data range, it effectively identified trends.

Sudden market fluctuations and external factors (e.g., economic policies, global events) sometimes impacted prediction accuracy.

The quality of predictions depends heavily on the dataset's size and quality.

Incorporating additional features like sentiment analysis from stock-related news can improve prediction accuracy.

2. User Feedback:

During testing, users appreciated features such as prediction visualization and the ability to track favorite stocks. However, some users suggested:

Adding notifications for stock updates.

Providing more granular controls for selecting prediction intervals.

3. Scalability:

The app is designed to support a growing number of users and features. It can be further enhanced by:

Integrating real-time stock price updates.

Expanding news coverage and providing advanced filtering options.

LIMITATIONS OF PROPOSED SYSTEM

Despite the successful implementation of the Stock Prediction Web App, several limitations constrain its overall performance and utility:

Prediction Accuracy:

The LSTM model cannot perfectly predict stock prices due to the inherent volatility of the stock market. There is often an offset between the predicted price and the actual market price, especially during periods of significant market turbulence or unexpected events.

Limited Prediction Horizon:

The current model is designed to predict stock prices one day ahead, based on the past 100 days of historical data. It cannot forecast long-term trends or prices for multiple days in the future, limiting its application for users looking for extended forecasting.

External Factors:

The model does not consider external influences such as macroeconomic policies, political events, or global crises, which significantly affect stock prices. Integration of such factors into the model requires advanced feature engineering and additional data sources.

Data Dependency:

The accuracy and reliability of the predictions depend heavily on the quality and completeness of historical stock data. Missing or noisy data can reduce the effectiveness of the model.

Computational Overhead:

Training the model and generating predictions for large datasets can be resource-intensive and time-consuming, making it challenging for systems with limited computational power.

PROPOSED ENHANCEMENTS

To overcome the current limitations and further improve the Stock Prediction Web App, the following enhancements are proposed:

Extended Prediction Horizon: Modify the model to predict stock prices for multiple future days instead of a single day.

Sentiment Analysis Integration: Incorporate sentiment analysis of financial news and social media to capture market sentiment and include it as a feature in the prediction model.

Real-Time Updates: Integrate real-time stock price updates and news to provide users with the most current information. Enable live tracking of stock price changes alongside predictions.

Improved User Interface: Add advanced charting tools and visualizations for better user insights. Include notifications or alerts for significant price movements or news updates.

Mobile Application: Develop a mobile version of the app to enhance accessibility and user experience. Utilize push notifications to keep users informed of updates and predictions.

CONCLUSION

The Stock Prediction Web App is a robust and user-friendly platform that leverages machine learning to provide stock price predictions and financial insights. The system is designed to:

Empower users with data-driven predictions based on historical trends.

Enhance decision-making through features like stock news integration and favorite stock tracking.

While the project has achieved its primary objectives, it also highlights the challenges associated with predicting the highly volatile and complex stock market. The limitations, such as the offset in predictions and the single-day forecasting capability, provide areas for future improvement.

The proposed enhancements, including extended forecasting horizons, sentiment analysis integration, and real-time updates, pave the way for a more comprehensive and powerful application. By addressing these limitations and incorporating the suggested enhancements, the app has the potential to become a valuable tool for traders and investors.

In conclusion, this project demonstrates the potential of combining machine learning techniques with interactive web applications to simplify stock market analysis. It serves as a foundation for future development and innovation in the field of financial technology.

BIBLIOGRAPHY:

The following resources were referenced during the development of the Stock Prediction Web Application. These sources provided essential guidance on Python programming, Streamlit framework development, machine learning model building, and data handling techniques.

Python Documentation

Python Software Foundation. "Python Documentation."

Available at: <https://docs.python.org/3/>

Referenced for general programming practices, syntax, and built-in libraries.

Streamlit Documentation

Streamlit Inc. "Streamlit Documentation."

Available at: <https://docs.streamlit.io/>

Utilized to understand the Streamlit library's usage for web app components, widgets, session state management, and deployment.

TensorFlow Documentation

TensorFlow. "TensorFlow Documentation."

Available at: https://www.tensorflow.org/api_docs

Used for the implementation of the LSTM model, model training, and predictions for stock prices.

Scikit-Learn Documentation

Scikit-Learn. "Machine Learning in Python."

Available at: <https://scikit-learn.org/stable/>

Referenced for pre-processing methods, model evaluation metrics, and other machine learning functionalities.

Pandas Documentation

Pandas. "Pandas Documentation."

Available at: <https://pandas.pydata.org/pandas-docs/stable/>

Used for data manipulation, cleaning, and transformation during data preprocessing and model training.

Yahoo Finance API Documentation

Yahoo Finance. "Yahoo Finance API."

Available at: <https://www.yahoofinanceapi.com/>

Used as a data source for stock price data, historical data, and other financial information for analysis and prediction.

USER MANUAL

Overview

The Stock Prediction Web App is an interactive platform that provides stock price predictions using machine learning and displays related financial news. Users can manage their favorite stocks, view historical data, and access predictions through an intuitive interface.

1. Login and Registration Screens

Login Page:

- Purpose: Allows existing users to access their accounts.

Fields:

- Username: Must be an alphanumeric string.
- Password: Must be at least 6 characters long.

Validations:

- Username and password cannot be empty.
- Passwords are masked for security.

Registration Page:

- Purpose: Enables new users to create an account.

Fields:

- Full Name: Must contain alphabetic characters only.
- Email: Must follow a valid email format like example@domain.com.
- Username: Unique alphanumeric string.
- Password: At least 6 characters long with one special character.

2. Home Page Screen

Purpose: Acts as the landing page after login. Displays an introduction to the app and guides users on navigating through its features.

Features:

- Brief explanation of the app's purpose.
- Navigation sidebar for switching between pages (Profile, Stock Prediction, News).

3. Profile Page Screen

Purpose: Displays user information and favorite stocks.

Features:

- Shows the user's full name, username, and email.
- Lists favorite stocks added by the user.
- Provides a button to remove stocks from the favorites list.

4. Stock Prediction Page Screen

Purpose: Enables users to input a stock ticker and view predictions for the next day's closing price.

Features:

- Input field for stock ticker (e.g., AAPL, TSLA).
- Predict button to trigger the machine learning model.
- Visualization of historical stock prices and predictions through a chart.
- Button to add the stock to the user's favorites list.

Validations:

- Stock ticker must be in a valid format (uppercase, no spaces).

5. News Page Screen

Purpose: Displays the latest financial news for all stocks or a specific stock ticker.

Features:

- Input field for optional stock ticker (e.g., AAPL).
- Fetch News button to retrieve articles.
- Display list of articles with titles, publication dates, and short summaries.

Validations:

Stock ticker input must follow the valid format (uppercase, no spaces).

6. Common Features

Navigation Sidebar:

Purpose: Allows users to switch between Home, Profile, Stock Prediction, and News pages seamlessly.