STOCK MARKET ANALYSIS AND PREDICTION

A Project Report
Submitted in partial fulfillment of
the requirements for the award of the Degree of

BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

By
Akash Singh (50)
Aman Jaiswal (12)

Under the esteemed guidance of
Prof. Shraddha Parab

DEPARTMENT OF INFORMATION TECHNOLOGY

TOLANI COLLEGE OF COMMERCE (AUTONOMOUS)

(Affiliated to University of Mumbai)

MUMBAI, 400 093
MAHARASHTRA
2023 – 2024

TOLANI COLLEGE OF COMMERCE (AUTONOMOUS)

(Affiliated to University of Mumbai)

MUMBAI-MAHARASHTRA-400093

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the project entitled, "**Stock Market Analysis and Prediction**", is bonafied work of **Akash Singh** bearing roll no: **50** submitted in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATIONTECHNOLOGY from University of Mumbai.

Internal Guide **Coordinator**

External Examiner

Date: **College Seal**

ABSTRACT

The “STOCK MARKET ANALYSIS AND PREDICTION” using historical price data and basic machine learning techniques. The objective is to provide a concise yet effective method for individuals interested in understanding and forecasting stock trends. The analysis begins by collecting historical stock price data, typically including open, close, high, and low prices, along with trading volume. Basic statistical measures such as mean, standard deviation, and correlation are computed to gain insights into the stock's historical performance and its relationship with market indices.

Machine learning models, including linear regression and moving averages, are employed for prediction. These models utilize historical price data to make short-term projections, recognizing underlying trends and patterns. The accuracy of predictions is evaluated using metrics such as mean squared error or root mean squared error. To enhance predictive power, sentiment analysis of relevant news and social media data can be integrated. Sentiment scores reflecting positive, negative, or neutral opinions can offer valuable inputs into the forecasting models.

In conclusion, this study demonstrates a simplified yet effective methodology for stock market analysis and prediction. By combining historical price data and basic machine learning techniques, individuals can gain insights into stock trends and make informed decisions. While acknowledging the inherent uncertainty of financial markets, this approach serves as a valuable starting point for individuals looking to navigate the complexities of stock trading.

Keywords: stock market, analysis, prediction, historical data, trend analysis, price patterns, statistics, forecasting.

ACKNOWLEDGEMENT

It takes great pleasure to me to present project report on **"STOCK MARKET ANALYSIS AND PREDICTION"**.

I would like express my deepest thanks and gratitude to my project guide, **Prof. Shraddha Parab**, for her guidance and help during each step of the project.

I would like to express my thanks and gratitude to my parents and their utmost support during the academic year so that I can focus properly on my project. With proper coordination and full fledge cooperation among me and my guide, I was able to complete this project successfully.

I would also like to express gratitude and thanks to Co-ordinator **Prof. Bandita Singh** and all the teaching staff of I.T Department for their utmost help and support in the project. Finally, I wish to express profound thanks to all those who helped me in any way regarding the project.

AKASH SINGH

DECLARATION

I hereby declare that the project entitled, "**STOCK MARKET ANALYSIS AND PREDICTION",** done at Tolani College of Commerce, has not been in any case duplicated to submit to any other universities for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfillment of the requirements for the award of degree of **BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)** to be submitted as final semester project as part of our curriculum.

Signature of Students

(Akash Singh)

TABLE OF CONTENTS

SR.NO	TITLE	PAGE.NO
1	**INTRODUCTION** 1.1 Introduction 1.2 Background 1.3 Objective 1.4 Purpose, Scope and Applicability 1.5 Achievements	1-5
2	**SURVEY OF TECHNOLOGIES**	6-15
3	**SYSTEM ANALYSIS** 3.1 Problem Definition 3.2 Modular Description	16-17
4	**REQUIREMENT ANALYSIS** 4.1 Functional Requirement 4.2 Software and Hardware Requirement	18-20
5	**PLANNING AND SCHEDULING** **GANTT CHART**	21-22
6	**SYSTEM DESIGN** 6.1 Class Diagram 6.2 Sequence Diagram 6.3 Use Case Diagram 6.4 Activity Diagram 6.5 ER Diagram 6.6 Data Design 6.7 Data Integrity and Constraints	23-37
7	**IMPLEMENTATION AND TESTING** 7.1 Code and Code Efficiency 7.2 Levels Of Testing 7.3 Test Cases	38-94
8	**COST ESTIMATION**	95-98
9	**RESULT AND DISCUSSION** 9.1 Test Report 9.2 User Documentation	99-107
10	**CONCLUSION**	108-109
11	**FUTURE SCOPE**	110
12	**REFERENCES**	111

INTRODUCTION

1.1 Introduction:-

The stock market, often referred to as the heart of a nation's economy, is a dynamic arena where financial instruments, such as stocks and bonds, are traded. It serves as a platform for companies to raise capital and for investors to allocate their funds in pursuit of financial growth.

The ever-evolving nature of the stock market, influenced by a myriad of economic, political, and social factors, presents both opportunities and challenges for market participants. In this context, the utilization of data-driven approaches for stock market analysis and prediction has become increasingly crucial.

Over the years, advancements in technology have led to an unprecedented surge in the availability of financial data. This deluge of information, ranging from company financials and market indices to news sentiment and social media chatter, has provided a fertile ground for researchers, analysts, and investors to explore new ways of understanding and predicting stock market movements.

This research endeavors to delve into the realm of stock market analysis and prediction, aiming to unravel the methodologies and tools that empower market participants to make informed decisions.

The multifaceted nature of stock market behavior necessitates a holistic approach that encompasses various analytical techniques, from fundamental and technical analysis to cutting-edge machine learning algorithms. By examining the strengths and limitations of these approaches, this study seeks to illuminate the potential implications for investors, traders, and decision makers.

1.2 Background:-

With Yahoo Finance is a widely recognized and reputable financial website that offers a comprehensive set of tools and resources for investors and individuals interested in finance and stock market analysis.

Stock Information: Yahoo Finance provides detailed information about individual stocks, including real-time and historical stock prices, trading volumes, market capitalization, and other key metrics.

News and Analysis: The website offers a wealth of financial news and analysis articles, including coverage of market trends, company earnings reports, and economic developments..

Interactive Charts: Yahoo Finance offers interactive and customizable stock charts that allow users to analyze historical price movements, apply technical indicators, and draw trendlines.

Portfolio Tracking: Users can create and manage investment portfolios, track the performance of their holdings, and receive alerts about significant price changes.

Financial Statements: You can access a company's financial statements, including income statements, balance sheets, and cash flow statements, which are essential for fundamental analysis.

Stock Screeners: Yahoo Finance provides stock screeners that allow users to filter and search for stocks based on specific criteria, such as market capitalization, sector, or P/E ratio.

Economic Calendar: The website offers an economic calendar that highlights important economic events, such as government reports and earnings releases, which can impact the financial markets.

1.3 Objective:-

The objective of stock market analysis and prediction is multifaceted, aiming to provide investors, traders, and financial institutions with valuable insights and actionable information to make informed decisions in the stock market. Primarily, it seeks to analyze historical market data, financial metrics, and relevant economic indicators to forecast future price movements and trends in the stock market. By understanding market dynamics, identifying patterns, and evaluating risk factors, stock market analysis assists investors in optimizing their investment strategies, maximizing returns, and minimizing risks. Additionally, it serves as a tool for assessing market efficiency, valuing securities, and managing portfolio risk. Furthermore, stock market analysis contributes to economic forecasting, strategic decision-making, and regulatory compliance, playing a vital role in maintaining the stability and integrity of financial markets. Ultimately, the overarching objective of stock market analysis and prediction is to support long-term wealth creation and financial well-being for investors and stakeholders in the dynamic and ever-changing landscape of the stock market.

Furthermore, stock market analysis aims to uncover underlying patterns, trends, and correlations within market data, allowing investors to gain insights into market dynamics and investor sentiment. By understanding the factors driving market movements, investors can anticipate changes and adapt their strategies accordingly. Additionally, stock market analysis plays a crucial role in assessing the relative value of securities, helping investors identify undervalued assets or overvalued stocks. This enables investors to make strategic investment decisions based on a comprehensive understanding of market conditions and asset valuations.

Moreover, stock market analysis contributes to portfolio optimization by enabling investors to diversify their investments across different asset classes, sectors, and geographical regions. By analyzing the performance and risk characteristics of various investment options, investors can construct well-balanced portfolios that align with their investment objectives and risk tolerance. Additionally, stock market analysis assists investors in evaluating the effectiveness of their investment strategies and making adjustments as needed to optimize portfolio performance over time

1.4 Purpose, Scope and Applicability:-

1.4.1 Purpose:-

The purpose of stock market analysis and prediction is to provide investors, traders, and financial institutions with valuable insights and information to navigate the complexities of the stock market effectively. By analyzing historical market data, financial metrics, and economic indicators, stock market analysis aims to forecast future price movements and trends in the market. This enables investors to make informed decisions regarding buying, selling, or holding stocks, with the ultimate goal of maximizing returns and minimizing risks. Additionally, stock

market analysis serves as a tool for evaluating market efficiency, assessing the intrinsic value of securities, and managing portfolio risk. It also contributes to economic forecasting, strategic decision-making, and regulatory compliance in financial markets. Overall, the purpose of stock market analysis and prediction is to support investors in achieving their financial objectives and securing their long-term financial well-being in the dynamic and ever-evolving landscape of the stock market.

Furthermore, stock market analysis and prediction aid in identifying investment opportunities by uncovering undervalued or promising stocks that have the potential for growth. It also assists in understanding investor sentiment and market psychology, shedding light on the underlying factors driving market movements. Moreover, stock market analysis plays a crucial role in enhancing market transparency and fostering investor confidence by providing reliable information and market insights.

It helps investors tailor their strategies to suit their risk tolerance, investment goals, and market outlook. Furthermore, it enables investors to stay informed about industry trends, company performance, and macroeconomic developments that may impact the market.

1.4.2 Scope:-

Data Collection:
Sources of historical stock price data, financial statements, and market indicators. Data preprocessing and cleaning techniques.

Technical analysis: Chart patterns, indicators, and trend identification.

Fundamental analysis: Financial ratios, earnings reports, and company performance

evaluation.

Sentiment analysis: Monitoring news sentiment, social media trends, and their impact on

stocks.

Machine learning models: Algorithms for predictive modeling and pattern recognition.

Regression analysis: Establishing relationships between stock prices and influencing factors.

Macroeconomic indicators: Interest rates, GDP growth, unemployment rates.

Market sentiment: Behavioral influences on stock prices. Industry-specific factors: Sector trends, regulations, innovations.

Time Horizons: Short-term trading analysis. Long-term investment analysis.
Medium-term trend identification.

Tools and Software: Mention tools like Excel, Python, R, statistical packages, and data visualization

libraries. Ethical Considerations: Insider trading regulations.
Adherence to market regulations and ethical practices.

Limitations: Acknowledge the inherent uncertainty and unpredictability of stock markets. Address the potential risks and challenges of prediction.

1.4.3 Applicability:-

- Describe how the analysis is used in investment, trading, and financial decision-making.
- Highlight scenarios where accurate predictions can lead to profitable outcome

1.5 Achievements:-

- Successful predictions lead to profitable investment decisions.
- Consistently outperforming market benchmarks showcases effective analysis.
- Accurate predictions aid in identifying and mitigating potential risks.
- Diversification and hedging strategies are informed by data-driven insights.

- Precise predictions enable optimal timing for entering or exiting positions.
- Maximizing gains and minimizing losses through strategic execution.

- Analysis assists in optimizing asset allocation for balanced growth.
- Achieving an optimal risk-return profile to align with investor goals.

- Development of AI-driven models and algorithms for prediction enhances technological innovation.
- Contributes to the advancement of predictive analytics

SURVEY OF TECHNOLOGIES

Survey on Tools and Technologies Applicable For Website Development

Abstract:-

The survey explores the utilization of Python and Streamlit as tools and technologies for website development. Python, renowned for its versatility and extensive library support, offers web developers a robust programming language to create dynamic and interactive websites.

Streamlit, a Python library, simplifies the development process further by enabling the creation of web applications with minimal effort, thanks to its intuitive interface and real-time updates.

Python's Versatility: Python's adaptability for various applications, from data analysis and scientific computing to web development and automation, makes it an indispensable language in the tech world.

Streamlit's Simplicity: Streamlit's intuitive API empowers individuals with limited programming experience to create impressive data-driven web applications without extensive coding. Applications in Data Science: Python's extensive libraries, coupled with Streamlit's capabilities, have ushered in a new era in data science, enabling professionals to develop interactive dashboards, data visualizations, and machine learning prototypes with ease.

Efficiency and Collaboration: The synergy between Python and Streamlit accelerates project development, encourages collaboration between data scientists and developers, and facilitates the sharing of insights with stakeholders.

Challenges and Future Developments: While Python and Streamlit offer remarkable advantages, this survey also discusses potential challenges, such as scalability concerns and security considerations, and offers insights into ongoing developments in the Python and Streamlit ecosystems.

Introduction:-

In the realm of website development, Python and Streamlit have emerged as a powerful combination of tools and technologies for creating interactive and user-friendly web applications. Python, renowned for its simplicity and versatility, serves as the programming language backbone, allowing developers to build robust backend systems and integrate various libraries and frameworks seamlessly. Streamlit, on the other hand, is a dynamic web app framework designed specifically for data scientists and developers, enabling them to transform data scripts into web applications with minimal effort. Together, Python and Streamlit offer a compelling solution for website development, catering to both data-driven applications and general web projects with their efficiency and ease of use.

Python is a versatile and beginner-friendly programming language known for its readability and extensive libraries. It's often referred to as a "Swiss Army knife" for programmers because of its wide range of applications. Python is used in web development, data analysis, machine learning, artificial intelligence, and much more.

Key Python Features:

- Easy-to-learn syntax: Python's clean and concise code makes it accessible to beginners.
- Rich library ecosystem: Python offers numerous libraries like NumPy, Pandas, and Matplotlib for data analysis, and frameworks like Django and Flask for web development.
- Cross-platform compatibility: Python runs on various operating systems, making it highly adaptable.

Streamlit:

Streamlit is a Python library that simplifies the process of creating interactive web applications. It's designed for data scientists and engineers who want to turn data scripts into shareable web apps quickly and easily, without extensive web development knowledge.

Key Streamlit Features:

- Rapid development: With Streamlit, you can transform data scripts into interactive apps with just a few lines of Python code.

- Data visualization: Streamlit seamlessly integrates with popular data visualization libraries, making it simple to create charts, plots, and dashboards.
- Customization: While Streamlit is beginner-friendly, it also allows for advanced customization and integration with external tools and services.

Why Python and Streamlit Together:

When Python and Streamlit are combined, you have a powerful duo for creating data-driven web applications. Python provides the robust back-end functionality for data manipulation and analysis, while Streamlit offers a straightforward way to build user-friendly front-end interfaces.

In summary, Python is a versatile and widely-used programming language, while Streamlit is a Python library that streamlines the creation of interactive web applications. Together, they empower developers and data scientists to bring their data to life in an accessible and user-friendly manner, making complex tasks seem effortless. Whether you're a beginner or an experienced coder, Python and Streamlit are an excellent combination to explore for your next project.

Tools Used To Develop web App:-

Introduction of Web App:-

Developing a web application for stock market analysis and prediction in Python involves leveraging a variety of tools and technologies to create a robust and user-friendly platform. This one-page introduction provides an overview of the key tools commonly used in building such applications.

Python is a versatile, high-level programming language well-suited for web application development. Its simplicity, readability, and vast ecosystem of libraries make it a popular choice.

Web frameworks like Django, Flask, and FastAPI simplify the development of web applications. They offer features such as routing, templating, and database integration.

Python's data analysis libraries, including Pandas, NumPy, and SciPy, enable data manipulation, analysis, and statistical modeling. These are essential for processing stock market data.

Libraries like Matplotlib, Seaborn, and Plotly allow you to create interactive and informative data visualizations, such as price charts and trend analysis graphs. Building RESTful APIs with libraries like Django REST framework or Flask-RESTful allows for seamless communication between the front-end and back-end components of the web app.

Security is crucial. Tools like OAuth2, JWT (JSON Web Tokens), or built-in authentication modules in web frameworks help secure user accounts and data access. Platforms like AWS, Heroku, or Azure provide cloud hosting solutions for deploying web applications. Containerization with Docker simplifies deployment management.

Using version control systems like Git and platforms like GitHub or GitLab ensures collaborative development, code tracking, and easy project management. Tools like Prometheus and Grafana enable real-time monitoring of application performance and user engagement. Analytics tools help gather insights from user interactions.

Tools Information:-

The different tools are used to first of design the Web App which are:

Visual Studio : – Visual Studio Code (VS Code) is a popular, free, and open-source code editor developed by Microsoft. It's designed for a wide range of programming languages and offers a lightweight, customizable, and efficient development environment. With its extensive library of extensions, VS Code can be tailored to meet specific programming needs, from web development to data science. It features an integrated terminal, Git integration, and intelligent code completion. Its real-time collaboration capabilities, debugging tools, and support for various operating systems make it a versatile choice for developers, enabling efficient code writing, testing, and deployment in a user-friendly and highly productive environment.

Visual Studio Language Resources – Visual Studio is a popular integrated development environment (IDE) primarily used for software development in various programming languages. Visual Studio provides extensive language support for a wide range of programming languages.

Streamlit- Stremlit is an open-source Python library that enables developers to create web applications quickly and easily using Python scripts. With Streamlit, developers can build interactive and data-driven applications for tasks such as data analysis, machine learning, visualization, and more, without the need for web development expertise. Streamlit provides a simple and intuitive interface for creating user interfaces (UIs) directly within Python scripts, allowing developers to focus on their application's logic and functionality. It offers a wide range of built-in components and widgets for creating interactive elements, such as sliders, buttons, and charts, as well as support for integrating external libraries and frameworks. Streamlit applications can be deployed locally or in the cloud with minimal setup, making it accessible to developers of all skill levels. Overall, Streamlit simplifies the process of building web applications in Python, empowering developers to create powerful and engaging applications with ease.

Python Libraries :-Python boasts a vast ecosystem of libraries and modules that enhance its functionality across various domains. Here are some of the most commonly used Python libraries, categorized by their respective domains:

1. **Pandas**: A versatile library for data manipulation and analysis, featuring data structures like DataFrames and Series.
2. **NumPy**: Essential for numerical computing, providing arrays and mathematical functions for scientific applications.
3. **streamlit:** This is the Streamlit library, which you use to create web applications with Python.
4. **sqlite3:** This library provides an interface for working with SQLite databases in Python.
5. **datetime:** This module provides functions to work with dates and times in Python.
6. **yfinance:** This library allows you to fetch historical market data, including stock prices, from Yahoo Finance.
7. **plotly.graph_objs:** Plotly is a visualization library, and `graph_objs` is a module within Plotly used to create interactive charts and graphs.
8. **nselib.capital_market:** This library provides functionality for accessing data related to capital markets, such as stock market indices, from the National Stock Exchange (NSE) in India.
9. **nselib.derivatives:** This library provides functionality for accessing data related to derivatives markets, such as futures and options, from the National Stock Exchange (NSE) in India.
10. **streamlit_lottie:** This library allows you to embed Lottie animations, which are vector animations created with Adobe After Effects, into Streamlit apps.
11. **requests:** The requests library is used to send HTTP requests in Python, allowing you to interact with web services and APIs.
12. **textblob:** TextBlob is a library used for processing textual data, including tasks such as sentiment analysis, part-of-speech tagging, and text classification.

Technologies Used To Design and Develop Website :-

Website:-

Designing and developing a website using Python and Streamlit is an innovative approach that combines the power of Python for backend functionality with Streamlit for creating interactive and data-driven web applications

Advantages and Limitations of Website Technology:-

Advantages of Website:-

1. **Ease of Use:**

   Python is known for its simplicity and readability, making it an excellent choice for developers of all skill levels. Streamlit, built specifically for data apps, is designed to be user-friendly and quick to learn. Developers can create interactive web applications with minimal effort.

2. **Data Visualization**

   Python has a rich ecosystem of data visualization libraries like Matplotlib, Seaborn, Plotly, and Bokeh. Streamlit seamlessly integrates with these libraries, making it easy to create interactive data visualizations.

3. **Cross-Platform:**

   Python and Streamlit are cross-platform, meaning you can develop and deploy applications on various operating systems, including Windows, macOS, and Linux.

4. **Customization**

   While Streamlit simplifies web development, it also allows for customization. Developers can integrate custom CSS, JavaScript, and HTML when needed to tailor the application's appearance and behavior.

5. **Deployment Options:**

   Streamlit apps can be easily deployed on various hosting platforms, including cloud services like AWS, Heroku, and Streamlit Sharing. This simplifies the deployment process, making it accessible to a broader audience.

6. **Community**

   Python has a large and active community, resulting in a vast number of libraries and packages availab le for various tasks. Streamlit also has a growing community, and its creators regularly release updates and improvements.

Limitations of Website :-

1. Limited Front-End Control-

Streamlit abstracts much of the front-end development, which is great for simplicity but limits control over the look and feel of the application. Developers seeking highly customized designs may find Streamlit's capabilities limited

2. Scalability :

While Streamlit is excellent for prototyping and small to medium-sized applications, it may fa ce scalability issues for larger, more complex web applications. It's not designed for building massive enterprise-level web apps..

3. Performance:- Streamlit applications can be slower compared to more optimized web development frameworks, especially for real-time applications or applications with high concurrency requirements.

4. Limited Template Choices- Streamlit offers limited built-in templates for application layouts. If you require complex and highly customized layouts, you may need to invest more effort in building them.

5. Browser Compatibility: Streamlit apps may not work perfectly in all browsers, as they heavily rely on the browser's ability to render JavaScript**.**

Justification of selection of Technology:-

The selection of technology for stock market analysis is a crucial decision, as it can significantly impact the efficiency, accuracy, and capabilities of your analysis platform

1. Python as the Primary Language:
   Versatility: Python is a versatile language that can handle data manipulation, analysis, and visualization effectively. Its extensive library ecosystem makes it an excellent choice for handling stock market data.Innovative.

2. Streamlit for Web Application Development:

   Rapid Development: Streamlit simplifies web application development with Python, allowing for the quick creation of interactive dashboards and data-driven interfaces.
   User-Friendly: Streamlit's ease of use means that analysts and developers can quickly build user-friendly applications without extensive web development expertise.

3. Data Analysis

   Pandas and NumPy: These libraries provide powerful tools for data manipulation, enabling the efficient handling of large datasets commonly found in stock market analysis.

4. Financial Data Integration:

   Alpha Vantage, Yahoo Finance API, or Quandl: These APIs provide reliable and up-to-date financial data for analysis and prediction.

5. Web Front-End with Streamlit:

   Simplicity: Streamlit simplifies the creation of interactive web interfaces without the need for extensive front-end development skills**.**
   Flexibility: It allows for the integration of Python code directly into web apps, enabling real-time updates and user interaction**.**

6. Security and Scalability:

   By leveraging established cloud providers and following best practices, you can ensure the security

and scalability of your application.

7. Community and Documentation:

   Python, Streamlit, and associated libraries have active and supportive communities with extensive documentation, tutorials, and resources.

8. Customization

   Python's extensibility allows for custom solutions and integration with other tools and APIs as needed for specific stock market analysis requirement

Conclusion:-

The t selection of Python, Streamlit, and associated libraries for stock market analysis is justified due to their versatility, ease of use, robust data analysis capabilities, and the ability to quickly develop interactive web applications. These technologies empower analysts and developers to build effective and user-friendly tools for analyzing and predicting stock market trends and making informed investment decisions.

SYSTEM ANALYSIS

3.1 Problem Definition:- Spotting error message

Streamlit is a Python library used for creating web applications with minimal effort, and like any software development, you may encounter various error messages during the development process. Here's how you can approach spotting error messages in Streamlit:.

3.1.1 Run Your Streamlit App:-

First, make sure your Streamlit app is running. You can typically start it by running a Python script tha tcontains your Streamlit code,Replace your_app.py with the actual name of your Streamlit application file.

3.1.2 Check the Terminal/Console:-

The terminal or console where you started your Streamlit app will often display error messages when something goes wrong. Errors can include Python exceptions, syntax errors, and Streamlit-specific issues. Carefully read through the error message to understand what went wrong.

3.1.2 Use Try-Except Blocks:-

In your Streamlit code, you can use try-except blocks to catch and handle errors gracefully. This can help you provide more informative error messages to users.

3.1.3 Check Your Code:-

Review your Streamlit code for any syntax errors, missing imports, or logical mistakes. Common issues include incorrect variable names, incorrect indentation, or missing brackets.

3.1.4 Debugging Tools:-

You can use debugging tools like 'pdb' (Python Debugger) to step through your code and identify issues. You can add breakpoints and inspect variables to understand what's happening at different stages of your application.

3.1.5 Review Dependencies:-

Sometimes, errors can be caused by incompatible library versions or missing dependencies. Make sure to check your virtual environment or package manager (e.g., pip or conda) to ensure all required packages are installed and up to date.

Modular Description:

- **Homepage :** The landing page of the website, providing an overview of the site.
- **Stock Quote:** Provides detailed information about a specific stock.
- **Market News:** Provides up-to-date news and articles related to the stock market..
- **User Accounts:** Allows users to create and manage their accounts.
- **Stock Screener:** Helps users filter and find stocks based on specific criteria.
- **Trading Tools:** Offers tools and resources for traders and investors.
- **Market Data API:** A backend module that collects and serves real-time market data.
- **Security and Privacy:** Ensures the security of user data and transactions.
- **Market Data API:** A backend module that collects and serves real-time market data.

REQUIREMENTS ANALYSIS

4.1 Functional Requirement: Authentication

- **Login-** The user can login to the authentication system username and password.
- **Logout-** The user can log out from the apps after complete their work.
- **Login failure-** If the user does not match in the login phase or the user has not yet being authorized by the user.

Non-Functional Requirement:

1. Performance Requirement

Performance requirements define the acceptable response times for system functions: -

1. The system shall take initial load time depending on the internet connection strength which also depends on the media from which the application is running.
2. The performance shall depend on the hardware components of the client.
3. The Stock market website must be accessible and up and running 24 hours a day, 7 days a week and 365 days a year.
4. The project shall display clear human-readable error messages.

2. Maintainability Requirement

It should be easy to add, remove or modify modules in this website. Debugging should not be difficult.

3. Availability Requirement

The website should be available 24 x 7. Services should be provided to the customers as and when requested.

4.2 Software and Hardware Requirement:-

4.2.1.1 Hardware Requirement:-

For development In System:	
RAM	Minimum 4 GB
Processor	Core2Duo Or Above
Space Required	3
Version	All device

4.2.1.1 Hardware Requirement table

4.2.2 Software Requirement:-

For development of Application in Systems:	
Operating System	Windows 10 or Above,
Front-End Language	Python development language, HTML, CSS & JS
Back-End Database	SQLite

4.2.2 Software Requirement table

Preliminary Product Description:-

The first step in the system development life cycle is the preliminary description to determine the feasibility of the system. The purpose of the preliminary description is to evaluate project request. It is not a design study nor does it include the collection of details to describe the business system in all respect.

Rather, it's the collecting of information that helps committee members to evaluate the merit of the project request and make an informal judgement about the feasibility of the proposed project.

Analysts working on the preliminary investigation should accomplish the fallowing objectives:-

- Clarify and understand the project request.
- Determine the size of the project.
- Assess costs and benefits of alternative approaches.
- Report the findings to management, with recommendations outlining the Acceptance or rejection of the proposal.
- Determine the technical and operational feasibility of alternative approaches.

Planning and Scheduling:-

Planning:-

The purpose of Project Planning is to identify the scope of the project, estimate the work involved, and create a project schedule. Project planning begins with requirements that define the software to be developed. The project plan is then developed to describe the tasks that will lead to completion.

Scheduling:-

The project schedule is the tool that communicates what work needs to be performed, which resources of the project members will perform the work and the timeframes in which that work needs to be performed. The project schedule should reflect all of the work associated with delivering the project on time

Task No	Task Name	Start Date	End Date	Duration
T1	Requirement Gathering	5-Aug-23	25-Aug-23	20
T2	Requirement Analysis	25-Aug-23	10-Sep-23	16
T3	Design	10-Sep-23	28-Sep-23	18
T4	Coding	28-Sep-23	10-Nov-23	44
T5	Testing	10-Nov-23	30-Nov-23	20
T6	Deployment	30-Nov-23	21-Jan-24	21
T7	Implementation	21-Jan-24	4-Feb-24	14

1.1. Planning and scheduling table:

Gantt chart:

A Gantt chart, commonly used in project management, is one of the most popular and useful ways of showing activities (tasks or events) displayed against time. On the left of the chart is a list of the activities and along the top is a suitable time scale.

Each activity is represented by a bar; the position and length of the bar reflects the start date, duration and end date of the activity. This allows you to see at a glance.

- What the various activities are
- When each activity begins and ends
- How long each activity is scheduled to last
- Where activities overlap with other activities, and by how much
- The start and end date of the whole project

To summarize, a Gantt chart shows you what has to be done (the activities) and when (the schedule).

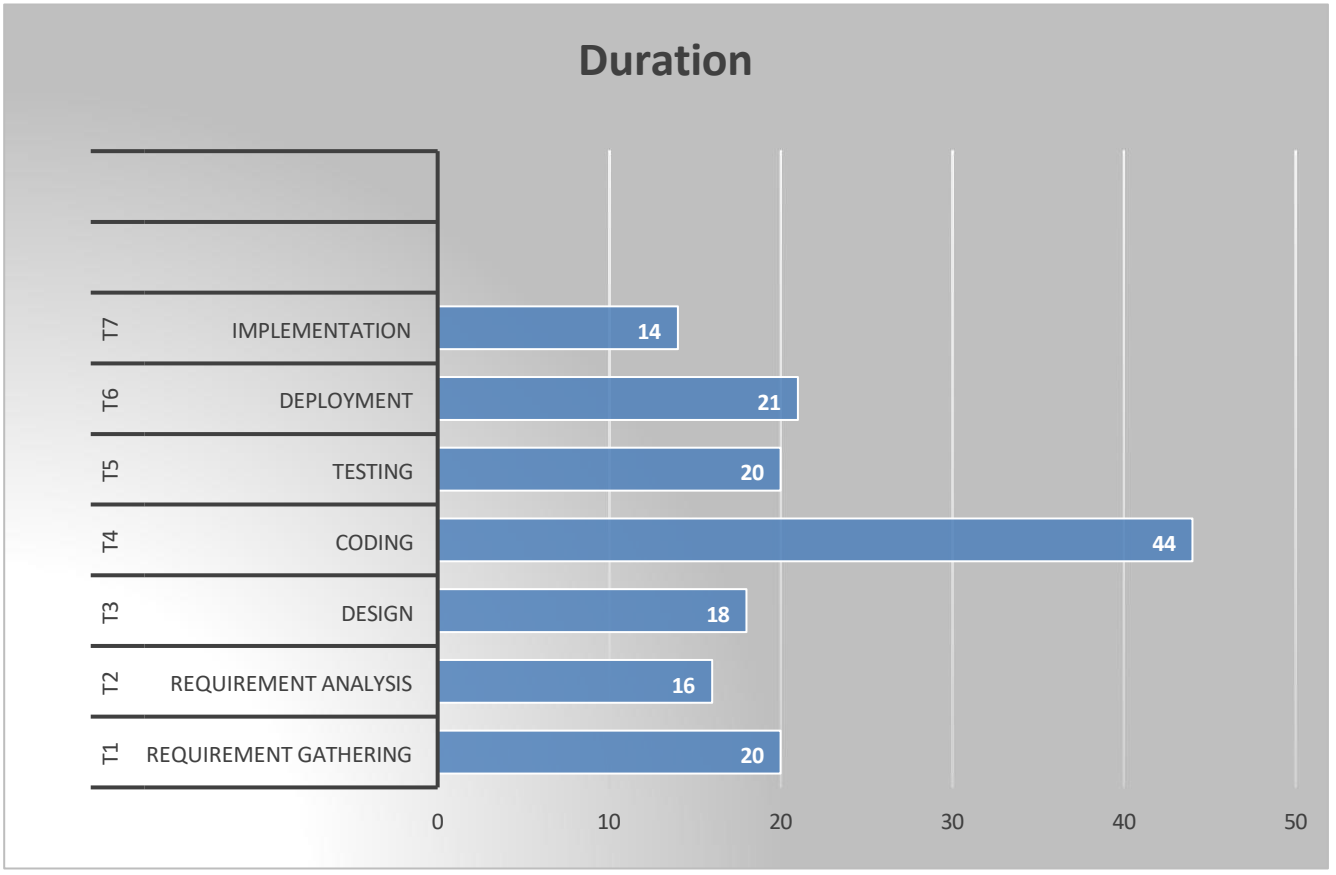


Fig: - 1 Gantt Chart for project Schedule Task against No of Days

SYSTEM DESIGN

SYSTEM DESIGN:-

System design is the process of defining the elements of a system such as the architecture, modules and components, the different interfaces of those components and the data that goes through that system. It is meant to satisfy specific needs and requirements of a business or organization through the engineering of a coherent and well-running system.

Following diagrams are:

- Class Diagram
- Sequence Diagram
- Use case Diagram
- Activity Diagram
- ER Diagram

1. Class Diagram:-

The Stock Market system class diagram describe the structure of a Doctor Appointment system classes, of their attributes, operation (or method), and the relationship among objects. The main classes of the Doctor Appointment System are Doctors, Login, Selection, Blood, Admin, User, and Appointment.

- **User: -** Represents individuals or entities who interact with the stock market system.
- **Market Data Fetcher: -** Represents components responsible for fetching real-time market data, including price updates, volume, and other relevant information.
- **Prediction Model : -** Represents models used for predicting stock market trends or outcomes.
- **Analysis Report:-** Represents reports generated based on stock market data and analysis.
- **Portfolio:-** Represents a collection of stocks and other assets held by an investor or trader
- **Stock :-** Represents a particular stock or security available for trading in the stock market.

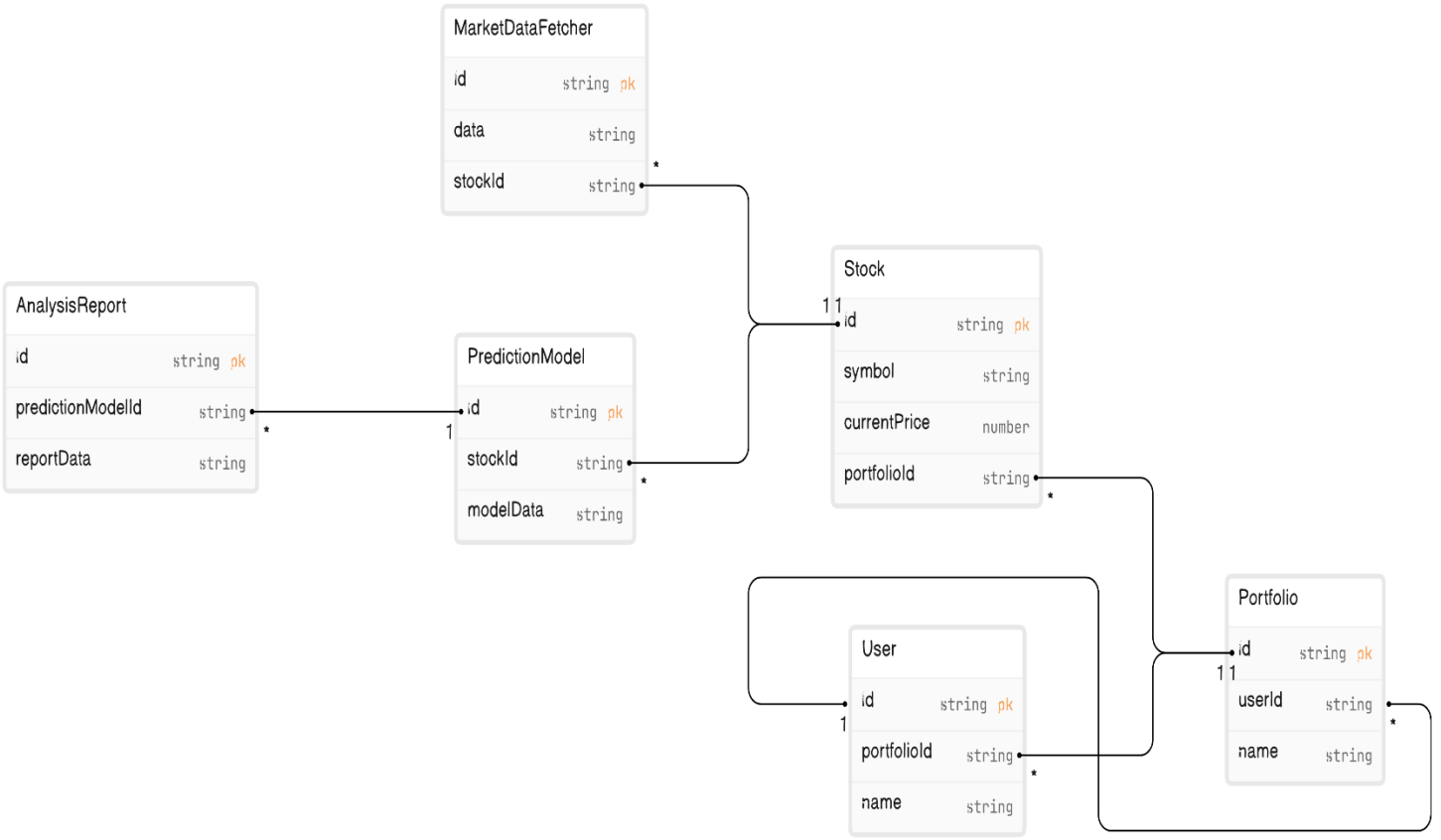


Fig: - 1 Class Diagram of Stock Market

2. Sequence Diagram:-

A UML Sequence Diagram for a Stock Market System describes the interactions and sequences of messages between objects or components in the system. Here's a high-level description of the elements and interactions you might depict in a UML Sequence Diagram for a stock market system.

The various objects in the Login Object , Forgot Password Object, Verification Object Database Object , Authentication Object interact over the course of the sequence, and user will not be able to access this page without verifying their identity.

The instance of class objects involved in this UML Sequence Diagram of Stock Market System are as Follows:-

- Login Object
- Verification Object
- Database Object
- Authentication Object

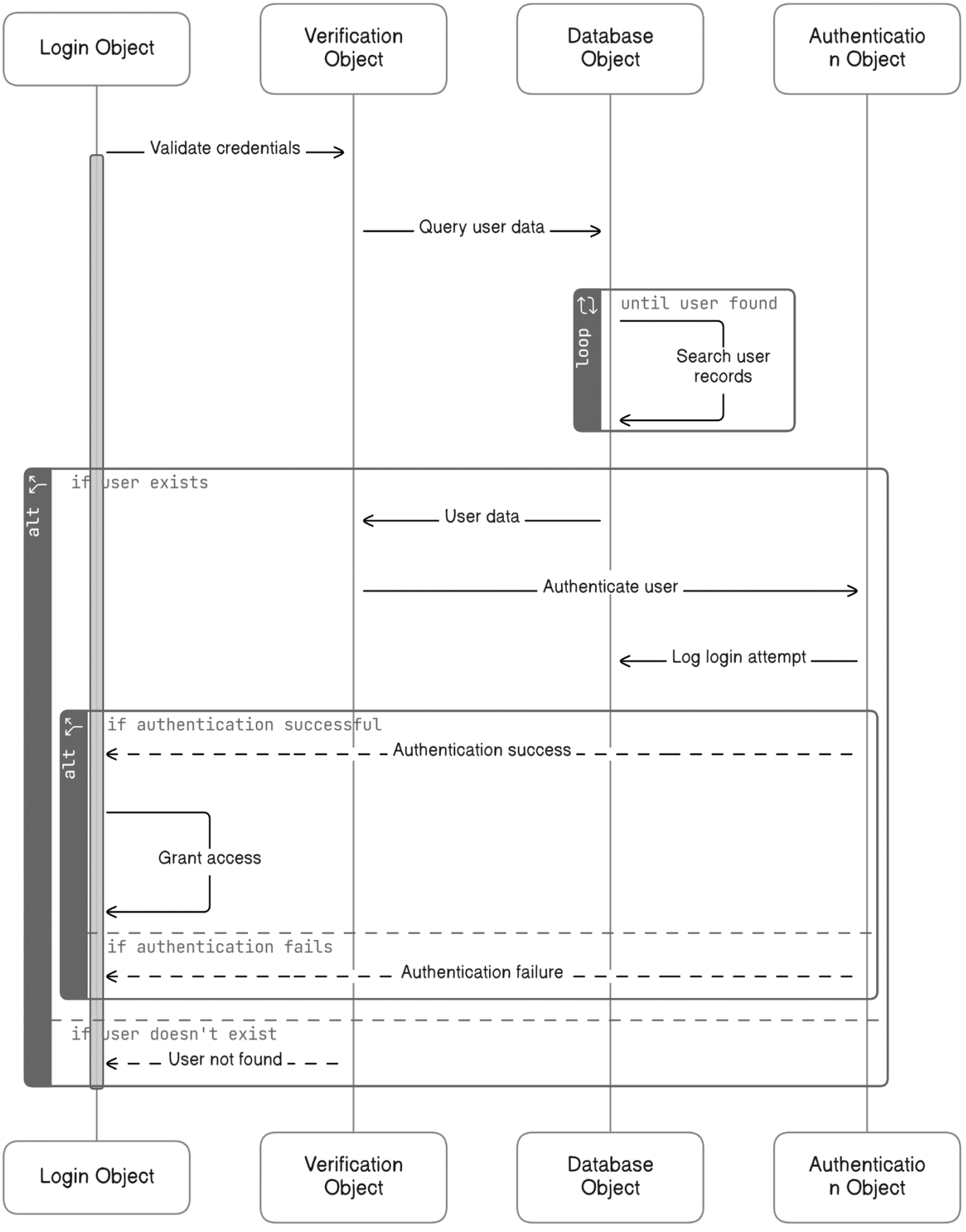


Fig: -2 Sequence Diagram of Stock Market

3. Use case Diagram:-

A use case diagram for a stock market system illustrates the different interactions and functionalities within the system, involving various actors and use cases. Here's a description of the main elements you might find in a stock market use case diagram:

A use case diagram for a stock market system provides a concise visual representation of the system's key functionalities and how different actors interact

Major element of the UML use case diagram of Stock Market System is shown on the use case diagram:-

The relationship between and among the actor and the use cases of Stock Market System:-

- **Admin Entity: -** Use case of System is Manage User Account , Manage prediction model and Fetch stock data , generate Analysis report.

- **User Entity: -** Use case of System is Manage login Account , Manage register and Request stock prediction report, view analysis report

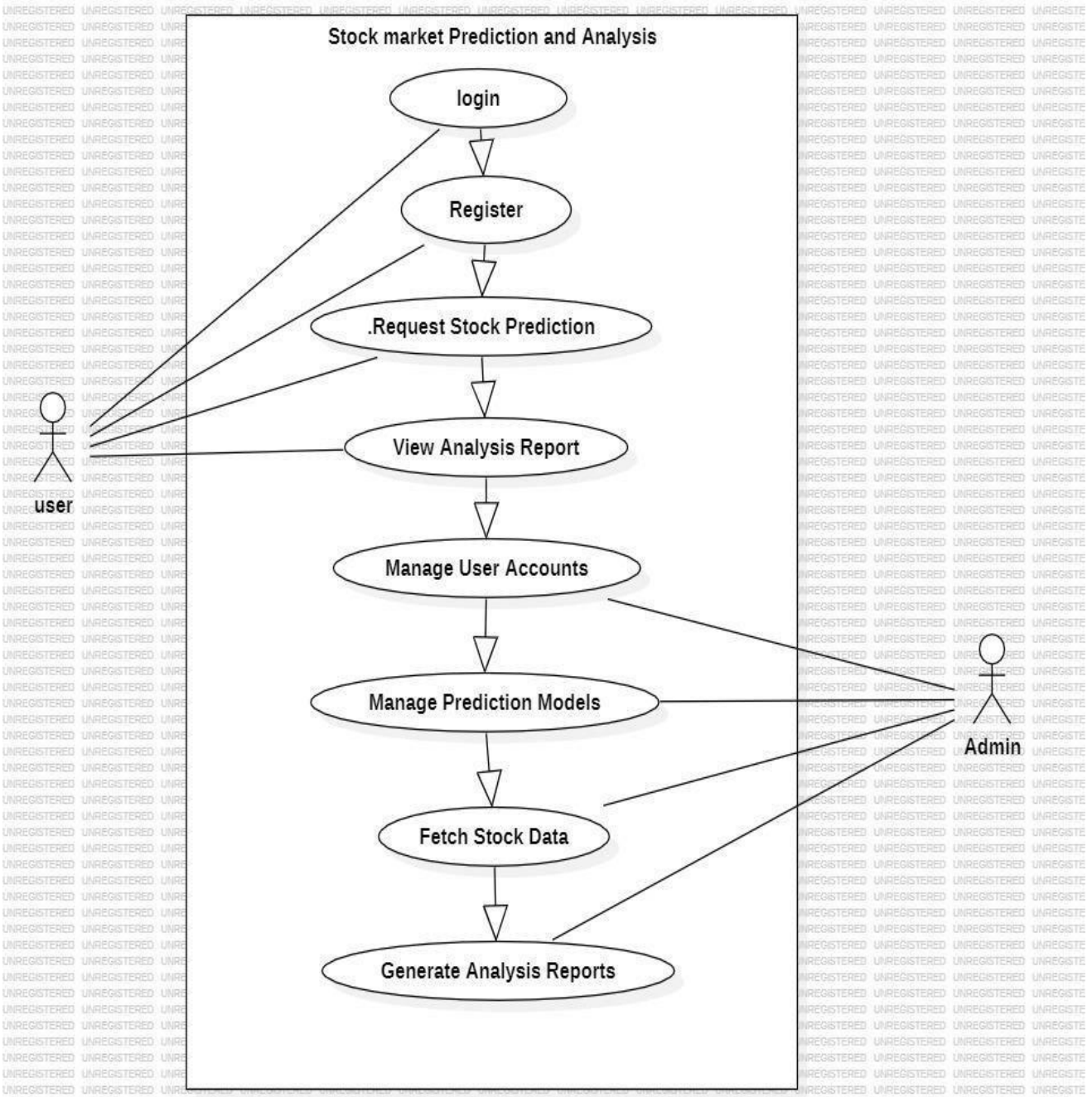


Fig: - 3 Use Case Diagram of Stock Market

4. Activity Diagram:-

This is the **Activity UML diagram of Doctor Appointment System** which show the flow between the activities of Test, schedule, Doctor, Appointment, Blood. The main activity involve in this UML Activity Diagram of Doctor Appointment System are as follows:

- Login activity
- Request data from various stock API Activity
- Load data from dataset Activity
- Trail Activity
- Predict price Activity
- Predict Stock Activity

❖ Features Of the Activity UML Diagram of Stock Market system:-

- In his Login Activity, This activity represents the process of user authentication and login to the system. Appointment is secure and user can access these page after login.
- Request Data from Various Stock API Activity: This activity involves sending requests to various stock market APIs or data providers to retrieve real-time or historical stock market data.
- Load Data from Dataset Activity:This activity involves loading historical stock market data from a dataset or database.
- Trail Activity:The "Trail Activity" seems to represent a trial or testing phase, where various stock market strategies or models are evaluated and tested.
- Predict Price Activity:This activity represents the process of predicting stock prices based on historical data or other factors.
- Predict Stock Activity:This activity focuses on predicting the future performance of specific stocks or assets.

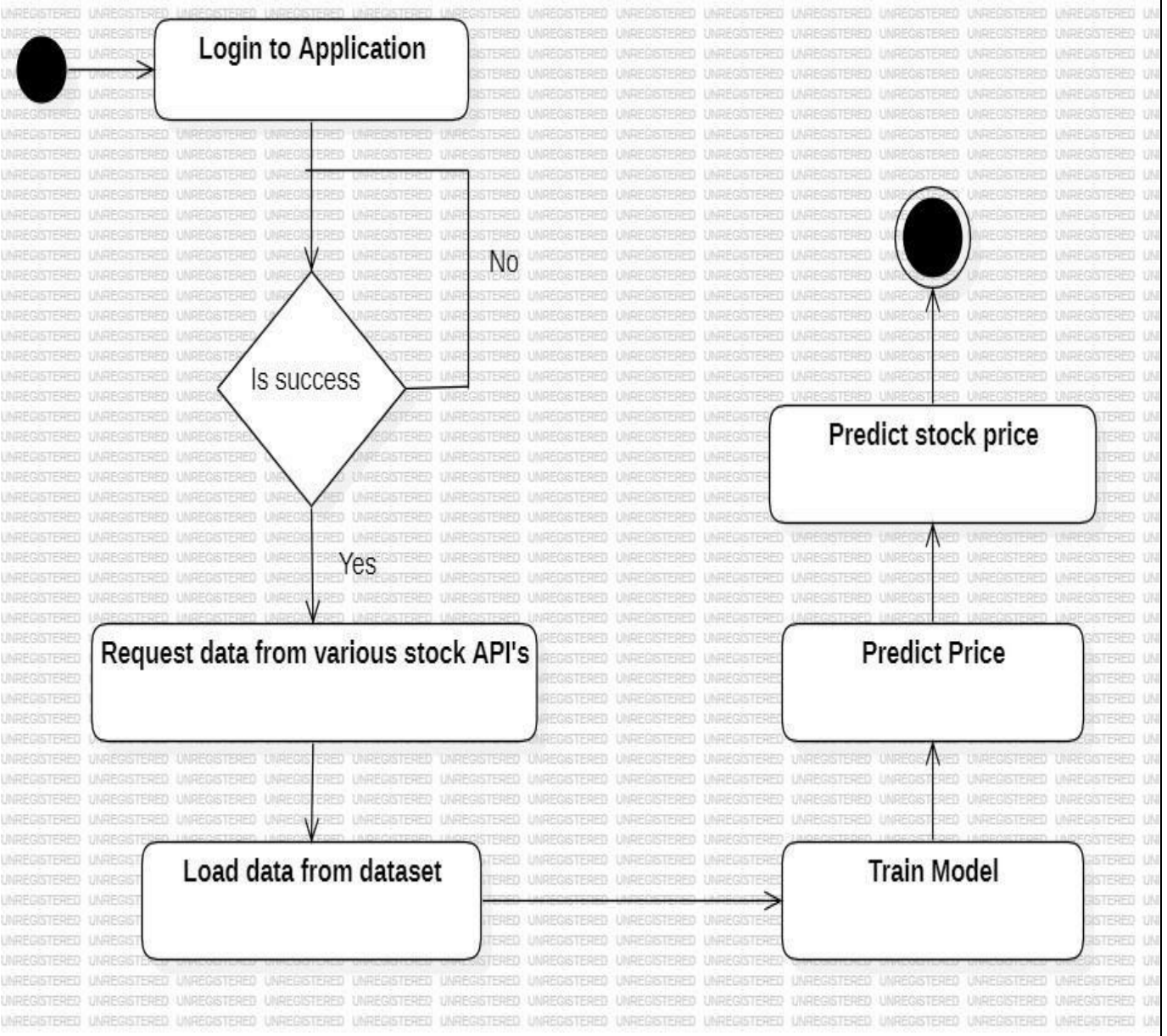


Fig: - 4 Activity Diagram of Stock Market

5 ER Diagram:-

The ER (Entity Relationship) Entity-Relationship (ER) diagram for a Stock Market Analysis and Prediction system is a complex task that would typically involve multiple entities, relationships, and attributes of Stock Market System functionalities. The Main entities of the Stock Market Analysis and Prediction system Doctointment System entities and their attributes:

5.1 **User: -** Attributes: UserID (Primary Key), Username, Password, Email, Role.

5.2 **Stock: -** Attributes: StockID (Primary Key), StockSymbol, CompanyName.

5.3 **StockData:** - Attributes: DataID (Primary Key), StockID (Foreign Key), Date, OpenPrice, ClosePrice, HighPrice, LowPrice, Volume.

5.4 **PredictionModel:-** Attributes: ModelID (Primary Key), ModelName, Description.

5.5 **Prediction:-** Attributes: PredictionID , ModelID .StockID , Date, PredictedPrice.

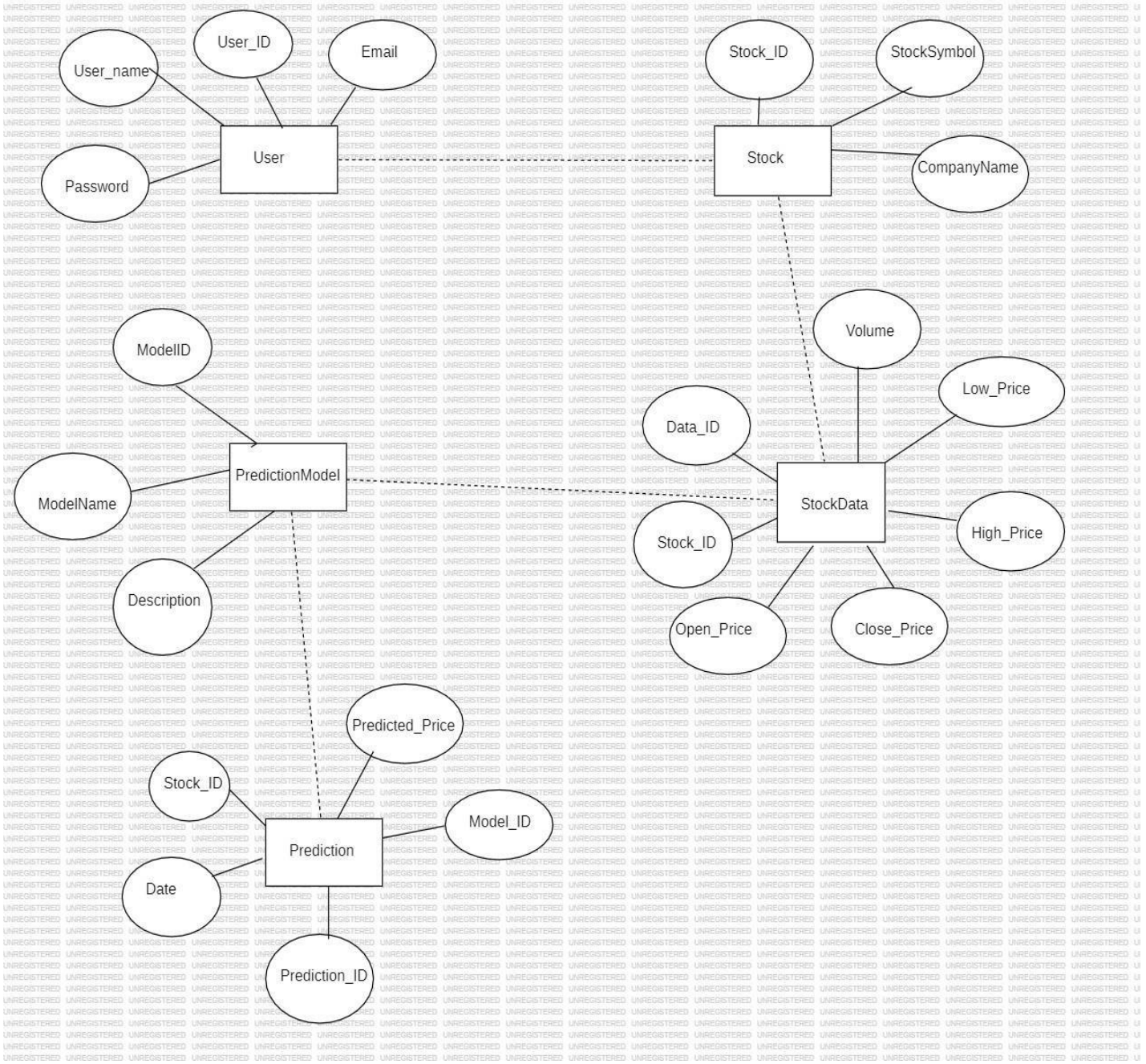


Fig: - 5 ER Diagram of Stock Market

6 Data design:

Data design for stock market analysis and prediction involves structuring and organizing various types of data to facilitate effective analysis, modeling, and forecasting of stock market trends and movements. This process encompasses the selection of relevant data sources, data collection, preprocessing, feature engineering, and model development. Here's a paragraph outlining key aspects of data design for stock market analysis and prediction:

"In stock market analysis and prediction, data design plays a crucial role in enabling the extraction of valuable insights from vast amounts of financial data. The process begins with the identification and acquisition of diverse data sources, including historical market data, company financials, economic indicators, and sentiment data from news articles and social media. These data are then collected, cleaned, and preprocessed to ensure accuracy, consistency, and completeness. Feature engineering techniques may be applied to transform raw data into meaningful features that capture relevant market dynamics and patterns. Additionally, advanced data analysis methods, such as time series analysis, statistical modeling, and machine learning algorithms, are employed to uncover relationships and trends within the data. The resulting predictive models can then be used to forecast stock prices, identify trading opportunities, and assess investment risks. Overall, effective data design is essential for enabling robust and reliable stock market analysis and prediction, empowering investors and traders to make informed decisions in dynamic financial markets."

Furthermore, in data design for stock market analysis and prediction, it's crucial to consider the quality and reliability of the data being used. This involves performing data validation and ensuring that the data is free from errors, outliers, and inconsistencies. Additionally, data design encompasses the integration of different data sources and formats, including structured data from databases, unstructured data from text sources, and semi-structured data from APIs or web scraping.

Moreover, data design involves establishing data pipelines and workflows to automate the process of collecting, processing, and updating data in real-time. This ensures that analysts and traders have access to the most up-to-date information to make timely decisions. Data governance practices, such as data security, privacy, and compliance with regulatory requirements, also play a vital role in data design for stock market analysis.

7 Data Integrity and Constraints:

Data integrity and constraints:

- ❖ Constraints
- ❖ GUI is only in English.
- ❖ Login and password is used for identification of user and there is no facility for guest.
- ❖ Data Integrity

1) Entity integrity
2) Referential integrity
3) Domain integrity

Definition of data integrity:

Data integrity in the stock market refers to the trustworthiness, accuracy, and reliability of financial data related to securities trading, market transactions, and corporate disclosures. It encompasses the assurance that the data accurately reflects the true state of the market, including stock prices, trading volumes, corporate earnings, and other relevant information. Maintaining data integrity is essential for ensuring fair, transparent, and efficient trading, as well as fostering investor confidence and trust in the financial markets.

1) Entity Integrity :

   Entity integrity ensures that each row in a database table has a unique identifier, typically through the use of primary keys, preventing duplicate records and maintaining data consistency.

2) Referential integrity :

   Referential integrity ensures that relationships between tables in a relational database are maintained, typically enforced through foreign key constraints, ensuring that references between related tables remain valid, preventing orphaned or inconsistent data.

3) Domain integrity :

   Domain integrity ensures that data values stored in a database adhere to predefined rules or constraints, such as data type, format, and range restrictions, enforcing the validity and consistency of data at the attribute level.

Data integrity and constraints:

Signup:

Signup	Data type	Constraints	Size
Username	Varchar	Primary Key	20
Email	Varchar	Unique, Not Null	25
Password	Varchar	Not Null, Check	8
Confirm Password	Varchar	Not null	8

Login:

Login	Data type	Constraints	Size
Email	Varchar	Unique, Not Null	25
Password	Varchar	Not Null, Check	8

Database:

Login	Data type	Constraints	Size
id	Int	Primary Key	
Username	Varchar	Primary Key	20
Email	Varchar	Unique, Not Null	25
Password	Varchar	Not Null, Check	8

Algorithms Design:-

STEP 1: Start

STEP 2: Login authenticate the user's credentials.

STEP 3: Enter valid username & password ensure the user provides correct login credentials.

STEP 4: If password and username are correct proceed to the trading platform.

STEP 5: Else display an error message and prompt the user to re-enter login credentials.

STEP 6: Trading Platform display market data including stock prices, trading volumes, and other relevant information.

Allow users to place buy or sell orders for stocks.

STEP 7: Repeat Steps 4-8 Users can continue to place orders or perform other trading activities as desired.

STEP 8: Logout End the trading session.

Security Issues :

Streamlit is a Python library used for creating web applications with simple and intuitive interfaces. If you encounter security-related errors while using Streamlit, there are a few common issues and simple ways to address them:

1. Port Binding:
   When running your Streamlit app, ensure that you're not binding it to a port that requires elevated privileges (like ports below 1024). Use a port number higher than 1024 (e.g., 8501) to avoid permission errors.

2. Input Validation:
   Always validate user inputs to prevent injection attacks or unintended behavior. Streamlit offers various input widgets, such as text inputs, sliders, and dropdowns. You can enforce constraints on these inputs to ensure they meet expected criteria.

3. File Uploads:
   If your Streamlit app allows file uploads, ensure proper validation and handling of uploaded files to prevent security vulnerabilities like directory traversal attacks or execution of malicious code.

4. Environment Isolation:
   Consider running your Streamlit app in a controlled environment, such as a Docker container, to isolate it from the host system and minimize potential security risks.

5. HTTPS: If deploying your Streamlit app in production, consider enabling HTTPS to encrypt communication between clients and the server, thus enhancing security.

6. Authentication and Authorization:
   Implement authentication and authorization mechanisms if your Streamlit app requires restricted access or user-specific functionality.

7. Regular Updates: Regularly update Streamlit and its dependencies to ensure you're using the latest versions with potential security patches.

8. Third-party Dependencies: Be cautious when using third-party dependencies in your Streamlit app. Only use reputable libraries, and regularly update them to incorporate security fixes.

9. Code Review: Perform thorough code reviews to identify and mitigate security vulnerabilities in your Streamlit app's codebase.

10. Community Support: If you encounter a specific security issue with Streamlit, seek assistance from the Streamlit community or report the issue on their GitHub repository.

IMPLEMENTATION AND TESTING

Implementation Approaches

Planning is the most vital aspect of this project. To implement the following project we had to gradually take a procedural approach. The entire project was highly dependent on the technique of each and every implementation aspect. The Project is basically implemented through ‘Visual Studio Code 1.87’ and the language used in it was ‘Python’.

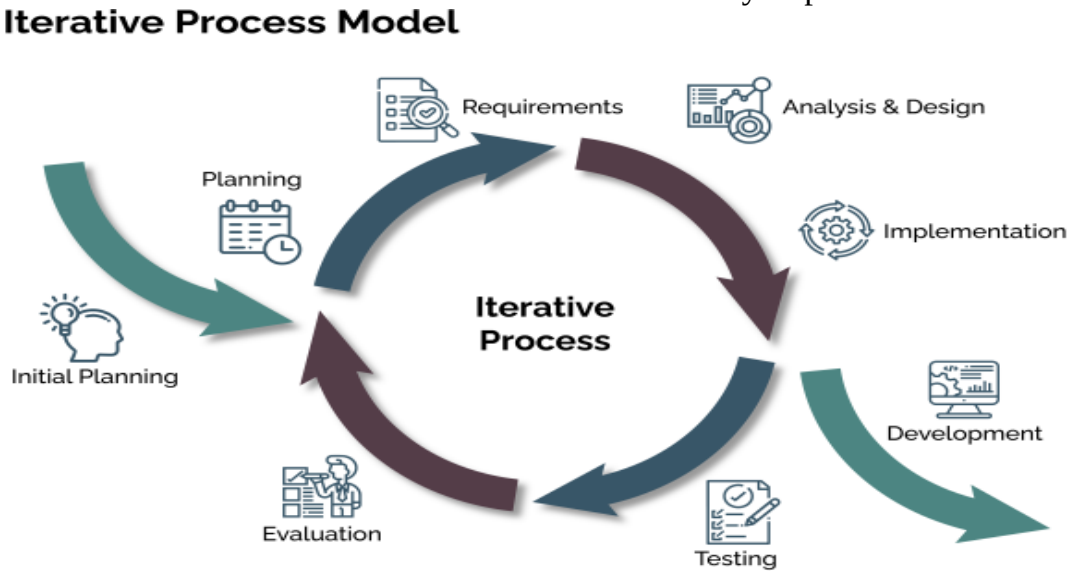
The Database Connectivity plays an important role in the back end of the project. The first one is a comprehensive Integrated Development Environment (IDE) tool for software development, while the second one is an Extension-based Code Editor. This project is entirely based on the System application and is made much convenient for the user to understand the application and its working procedures.

ADOPTED METHODOLOGY

Iterative Model :

The iterative model, commonly associated with software development, can also be effectively applied to stock market analysis, allowing analysts to refine their strategies, improve decision-making processes, and adapt to changing market conditions over time. In the context of stock market analysis, the iterative model involves a cyclical process of data collection, analysis, hypothesis formulation, testing, and refinement.

Initially, analysts start by defining their objectives and gathering relevant data from various sources, including historical stock prices, financial statements, economic indicators, and news articles. This data serves as the foundation for the analysis process.



What is iterative model?

The iterative model is a software development approach that emphasizes the repetition of a cyclical process of development, testing, and refinement. In this model, the software project is divided into smaller, manageable increments or iterations, with each iteration representing a complete cycle of development. The iterative model allows for flexibility and adaptability, as it enables the incorporation of feedback and changes throughout the development process.

Each iteration typically includes requirements gathering, design, implementation, testing, and evaluation stages. After completing an iteration, the software is reviewed, and feedback is gathered from stakeholders, users, or team members. Based on this feedback, adjustments and improvements are made to the software in subsequent iterations.

The iterative model is particularly well-suited for projects where the requirements are not fully understood or may evolve over time. It allows for early delivery of working software and enables stakeholders to see tangible progress at regular intervals. Additionally, the iterative model promotes collaboration and communication among team members, as they work closely together to deliver incremental improvements to the software.

Overall, the iterative model promotes a dynamic and iterative approach to software development, where continuous refinement and adaptation lead to the delivery of high-quality software that meets the evolving needs of stakeholders.

Another benefit of the iterative model is its ability to accommodate changes in project scope or requirements without significantly disrupting the development timeline. Because the project is divided into manageable iterations, adjustments can be made to the scope, priorities, or features between iterations based on evolving business needs or market dynamics. This flexibility helps ensure that the final product aligns closely with the stakeholders' expectations and delivers maximum value.

Additionally, the iterative model encourages frequent feedback loops with stakeholders, users, and domain experts, facilitating collaboration and ensuring that the software meets their needs effectively. By involving stakeholders in the iterative development process, the team can validate assumptions, gather insights, and make informed decisions about the direction of the project. This collaborative approach promotes transparency, trust, and accountability, ultimately leading to the successful delivery of high-quality software solutions.

The various phases of Iterative model are as follows:

In an iterative model, the development process is divided into several phases or stages, each of which involves iterative cycles of planning, execution, evaluation, and refinement. Here are the various phases typically associated with an iterative model:

1. Planning Phase:-
   Identify project objectives, requirements, and constraints. Define the scope of the project and establish priorities. Develop an initial project plan outlining tasks, timelines, and resource allocations.

2. Analysis Phase:-
   Gather and analyze user requirements, stakeholder feedback, and business needs. Identify key features, functionalities, and system capabilities. Create user stories, use cases, or functional requirements documents to capture system requirements.

3. Design Phase:-
   Develop an initial system architecture and high-level design based on the analysis findings. Define the overall system structure, components, and interfaces. Create prototypes, wireframes, or mock-ups to visualize the user interface and workflow.

4. Implementation Phase:-
   Translate the design specifications into working software components. Write code, develop algorithms, and implement system features according to the requirements. Conduct unit testing to verify the functionality and correctness of individual components.

5. Testing Phase:-
   Perform integration testing to ensure that different system components work together as intended. Conduct functional testing to validate that the system meets the specified requirements and user expectations. Identify and fix defects, bugs, or inconsistencies through iterative cycles of testing and debugging.

6. Evaluation Phase:-
   Review and assess the progress, quality, and performance of the developed system. Solicit feedback from users, stakeholders, and domain experts to evaluate the usability and effectiveness of the system. Measure key performance indicators (KPIs) such as reliability, scalability, and user satisfaction.

7. Refinement Phase:-
   Incorporate feedback and insights from the evaluation phase to refine and improve the system. Make iterative adjustments to the design, functionality, and implementation based on user preferences and evolving requirements. Prioritize enhancements and feature enhancements for future iterations based on their impact and feasibility

8. Deployment Phase:-
   Release the updated version of the system to users or stakeholders. Conduct user training and provide documentation to support adoption and use of the system. Monitor system performance and address any issues or concerns that arise during deployment.

9. Feedback and Iteration:-
   Gather feedback from users and stakeholders following deployment. Use feedback to inform future iterations and enhancements to the system. Repeat the iterative cycle of planning, analysis, design, implementation, testing, evaluation, refinement, and deployment as needed to continuously improve the system.

When to use the Iterative Model?

1. Uncertain or Evolving Requirements:- When the project requirements are not well-defined or are likely to change over time, the iterative model allows for flexibility and adaptation to evolving needs.
2. Complex Projects: For projects with complex functionalities, intricate system interactions, or diverse stakeholder requirements, the iterative model enables gradual decomposition of requirements into manageable increments, facilitating better understanding and management of complexity.
3. Customer-Centric Development: In projects where user involvement and satisfaction are crucial, such as consumer-facing applications or customer-centric businesses, the iterative model allows for continuous feedback and refinement based on user input, ensuring that the final product meets user expectations.
4. Highly Innovative Projects: When innovation, experimentation, or exploration is necessary, the iterative model provides a framework for iterative testing of new ideas, validation of assumptions, and refinement of concepts based on real-world feedback.

Advantage of Iterative Model:

1. Flexibility: The iterative model allows for flexibility and adaptability to changing requirements, enabling teams to respond quickly to feedback and evolving stakeholder needs.

2. Early Delivery of Value: Incremental development and delivery of functionality allow stakeholders to start realizing benefits sooner, even before the entire project is complete.

3. Continuous Improvement: By enabling continuous feedback and refinement, the iterative model promotes continuous improvement, ensuring that the final product meets user needs and quality standards.

4. Risk Reduction: Incremental development and testing help identify and mitigate risks early in the project lifecycle, reducing the likelihood of major issues or setbacks later on.

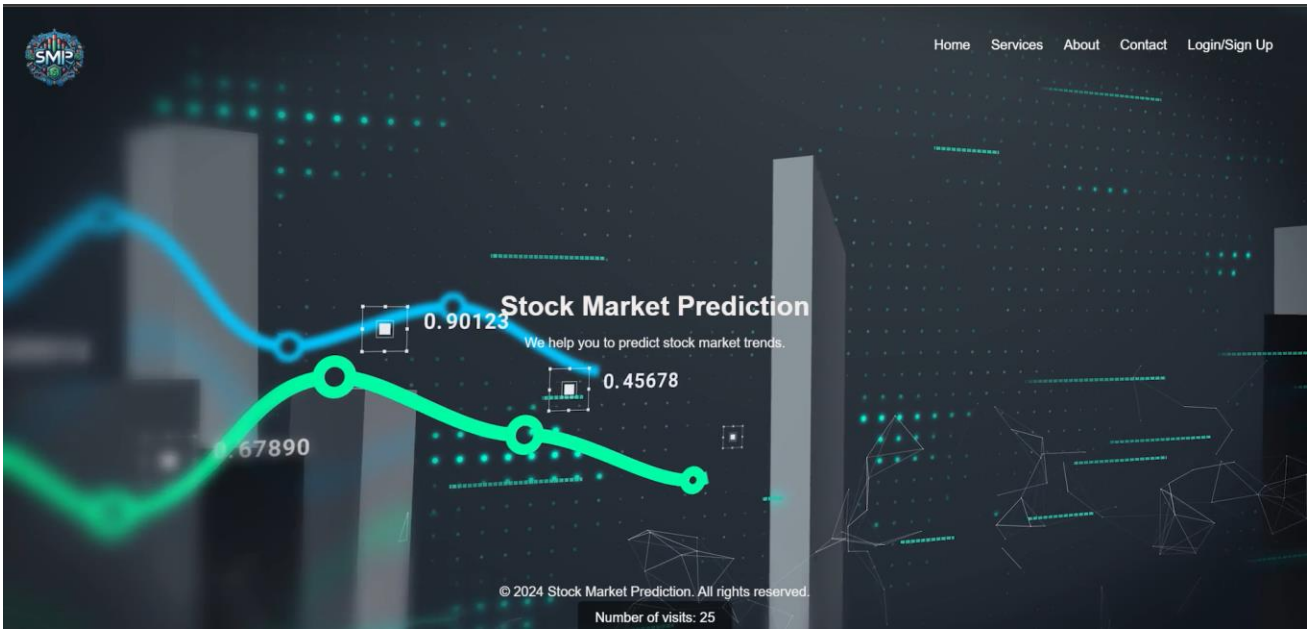
Disadvantage of Iterative Model:

1. Complexity Management: Managing multiple iterations and integrating new functionality can introduce complexity and overhead, requiring careful planning and coordination.
2. Increased Cost: The iterative model may result in higher development costs compared to traditional sequential approaches, particularly if extensive changes are required during each iteration.
3. Uncertain Project Duration: The iterative model makes it challenging to predict project duration accurately, as the scope and timeline may evolve based on iterative feedback and adjustments.

7.1 Coding details and Code Efficiency

Source Code:-

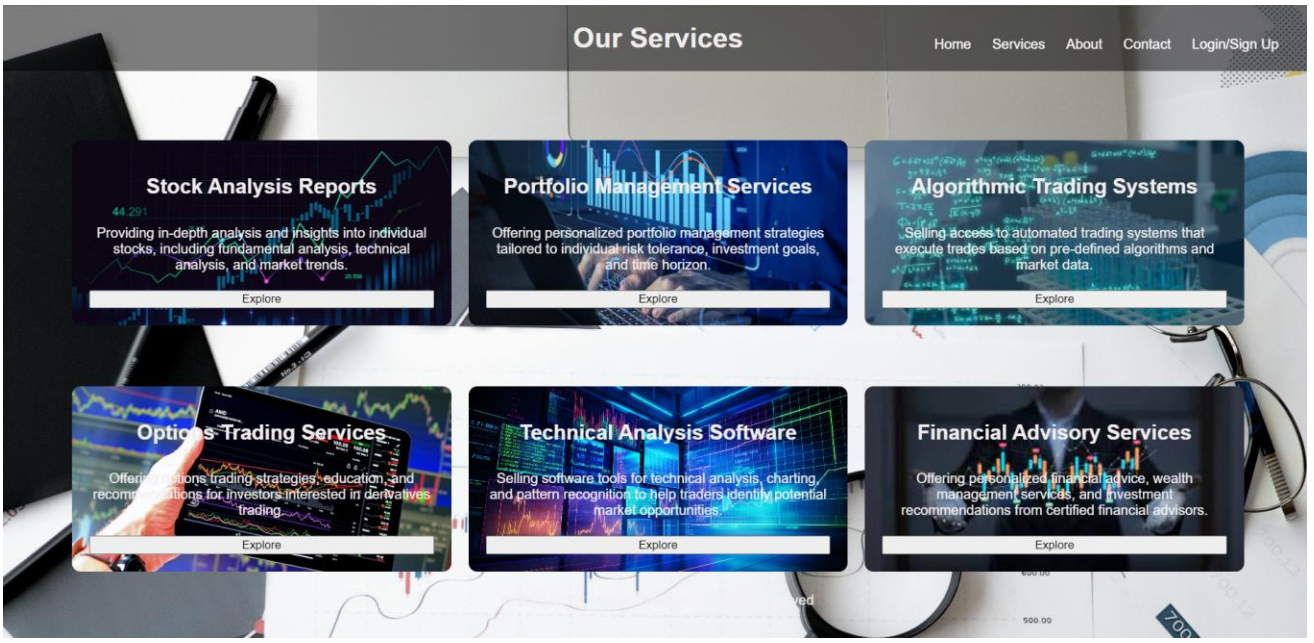
Stock.html :




```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Stock Market Prediction</title>
<link rel="icon" href="logo-Photoroom.png-Photoroom.png" type="image/x-icon">
<link rel="stylesheet" href="./styles.css">
<style>
    #visit-counter {
        position: fixed;
        bottom: 0px;
        left: 50%;
        transform: translateX(-50%);
        background-color: rgba(0, 0, 0, 0.5);
        color: #fff;
        padding: 10px 20px;
        border-radius: 5px;
        font-size: 16px;
    }
</style>
<script>
    // Check if localStorage is available
    if (typeof(Storage) !== "undefined") {
        // Check if the 'visits' key exists in localStorage
        if (localStorage.getItem('visits')) {
            // If the 'visits' key exists, increment its value by 1
```

```
            var visits = parseInt(localStorage.getItem('visits')) + 1;
            localStorage.setItem('visits', visits);
        } else {
            // If the 'visits' key does not exist, set it to 1
            localStorage.setItem('visits', 1);
        }
        // Display the number of visits on the page
        document.addEventListener("DOMContentLoaded", function() {
            var visitCounter = document.createElement("div");
            visitCounter.id = "visit-counter";
            visitCounter.textContent = "Number of visits: " + localStorage.getItem('visits');
            document.body.appendChild(visitCounter);
        });
    } else {
        // If localStorage is not available, display a message
        document.write("<p>Number of visits: localStorage not supported</p>");
    }
</script>
</head>
<body>

<video autoplay muted loop class="video-background">
  <source src="pexels-oleg-gamulinskii-18743334 (1080p).mp4" type="video/mp4">
  Your browser does not support the video tag.
</video>
<div class="logo">
  <img src="logo-Photoroom.png-Photoroom.png" alt="Your Logo">
</div>

<nav>
  <ul>
    <li><a href="#home">Home</a></li>
    <li><a href="./services.html">Services</a></li>
    <li><a href="./about.html">About</a></li>
    <li><a href="./contact.html">Contact</a></li>
    <li><a href="http://localhost:8501/">Login/Sign Up</a></li>
  </ul>
</nav>

<div id="home" class="content">
  <h1>Stock Market Prediction</h1>
  <p>We help you to predict stock market trends.</p>
</div>

<footer>
    <p>&copy; 2024 Stock Market Prediction. All rights reserved.</p>
</footer>

</body>
</html>
```

Services.html:-




```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Services</title>
    <link rel="icon" href="logo-Photoroom.png-Photoroom.png" type="image/x-icon">
    <link rel="stylesheet" href="styles.css">
    <style>
        body {
            background-image: url('services.jpg'); /* Replace 'services.jpg' with the path to your default background image */
            background-size: cover;
            background-repeat: no-repeat;
            margin: 0;
            padding: 0;
            font-family: Arial, sans-serif;
        }

        header {
```

```
    text-align: center;
    padding: 20px 0;
    background-color: rgba(0, 0, 0, 0.5);
    color: white;
}

header h1 {
    margin: 0;
}

nav ul {
    list-style: none;
    padding: 0;
}

nav ul li {
    display: inline;
    margin-right: 20px;
}

nav ul li a {
    color: rgb(255, 255, 255);
    text-decoration: none;
}

main {
    padding: 80px;
    display: grid;
    grid-template-columns: repeat(3, 1fr); /* Three equal columns */
    gap: 20px; /* Gap between grid items */
    opacity: 0;
    animation: fadeIn 1s ease forwards; /* Animation for grid items */
}
```

```
.service {
    background-size: cover;
    background-repeat: no-repeat;
    background-position: center center;
    border-radius: 10px;
    padding: 20px;
    color: white;
    display: flex;
    flex-direction: column;
    justify-content: space-between;
    text-align: center;
    animation: fadeInItem 1s ease forwards; /* Animation for individual grid items */
}

.explore-btn {
    background-color: #007bff;
    color: white;
    border: none;
    padding: 10px 20px;
    cursor: pointer;
    border-radius: 5px;
    transition: background-color 0.3s;
    align-self: center; /* Center the button horizontally */
}

.explore-btn:hover {
    background-color: #0056b3;
}

@keyframes fadeIn {
    from {
        opacity: 0;
    }
    to {
```

```
            opacity: 1;
        }
    }

    @keyframes fadeInItem {
        from {
            transform: translateY(50px);
            opacity: 0;
        }
        to {
            transform: translateY(0);
            opacity: 1;
        }
    }

</style>
</head>
<body>
    <header>
        <h1>Our Services</h1>
        <nav>
            <ul>
                <li><a href="./stock.html">Home</a></li>
                <li><a href="./services.html">Services</a></li>
                <li><a href="./about.html">About</a></li>
                <li><a href="./contact.html">Contact</a></li>
                <li><a href="http://localhost:8501/">Login/Sign Up</a></li>
            </ul>
        </nav>
    </header>

    <main>
        <section class="service" style="background-image: url('first.jpg');">
            <h2>Stock Analysis Reports</h2>
```

Providing in-depth analysis and insights into individual stocks, including fundamental analysis, technical analysis, and market trends.
<button onclick="purchaseService('Stock Analysis Reports', 499, 'https://seekingalpha.com/uploads/2017/4/2/4974797/Equity-Research-Report-Expedia.pdf')">Explore</button>
</section>

<section class="service" style="background-image: url('Portfolio Management Services.jpg');">
<h2>Portfolio Management Services</h2>
Offering personalized portfolio management strategies tailored to individual risk tolerance, investment goals, and time horizon.
<button onclick="purchaseService('Portfolio Management Services', 799, 'https://docs.google.com/spreadsheets/d/1KMMM3ZRWcEG3Z76SWEAhoM_ArKbpo2Lt/edit?usp=drive_link&ouid=105557778832099714272&rtpof=true&sd=true')">Explore</button>
</section>

<section class="service" style="background-image: url('Algorithmic Trading Systems.jpg');">
<h2>Algorithmic Trading Systems</h2>
Selling access to automated trading systems that execute trades based on pre-defined algorithms and market data.
<button onclick="purchaseService('Algorithmic Trading Systems', 999, 'https://drive.google.com/file/d/1qZpdOrvabmilVdNe8YbusmjKUlM_OhXv/view?usp=drive_link')">Explore</button>
</section>

<section class="service" style="background-image: url('Options Trading Services.jpg');">
<h2>Options Trading Services</h2>
Offering options trading strategies, education, and recommendations for investors interested in derivatives trading.
<button onclick="purchaseService('Options Trading Services', 599, 'https://docs.google.com/spreadsheets/d/1tVAQRPvJvfpxYP1eAQ956xMklAntmiVV/edit?usp=drive_link&ouid=105557778832099714272&rtpof=true&sd=true')">Explore</button>
</section>

```
        <section class="service" style="background-image: url('Technical Analysis Software 2.jpg');">
            <h2>Technical Analysis Software</h2>
            <p>Selling software tools for technical analysis, charting, and pattern recognition to help traders identify potential market opportunities.</p>
            <button onclick="purchaseService('Technical Analysis Software', 699, 'https://youtu.be/JSWlKipfCVo?si=DYC_TgQzIieyuNTv')">Explore</button>
        </section>

        <section class="service" style="background-image: url('Financial Advisory Service2.jpg');">
            <h2>Financial Advisory Services</h2>
            <p>Offering personalized financial advice, wealth management services, and investment recommendations from certified financial advisors.</p>
            <button onclick="purchaseService('Financial Advisory Services', 899, 'https://youtube.com/playlist?list=PLxNHpNhDaEFLL6j4Pe-uo2TQC0KAlSudh&si=kBzPLIgMiMXaKYN1')">Explore</button>
        </section>
    </main>

    <footer>
        <p>&copy; 2024 Stock Market Prediction. All rights reserved</p>
    </footer>

    <script src="https://checkout.razorpay.com/v1/checkout.js"></script>
    <script>
        function purchaseService(serviceName, servicePrice, pdfUrl) {
            var options = {
                key: 'rzp_test_1cfUrUFbSKhchN',
                amount: servicePrice * 100,
                currency: 'INR',
                name: 'Stock Market Prediction',
                description: `Purchase ${serviceName}`,
                image: 'logo-Photoroom.png-Photoroom.png',
                handler: function(response) {
                    console.log('Payment successful:', response);
```

```
                alert('Payment successful! Redirecting to PDF download...');
                // Redirect to the PDF URL after successful payment
                window.location.href = pdfUrl;
            },
            prefill: {
                name: 'John Doe',
                email: 'john@example.com',
                contact: '9876543210'
            },
            notes: {
                address: 'Razorpay Corporate Office'
            },
            theme: {
                color: '#007bff'
            }
        };

        var rzp = new Razorpay(options);
        rzp.open();
    }
</script>
</body>
</html>
```

about.py:-

Deploy

Stock Market Prediction

Choose an option:

About

About

About page content goes here.

Welcome to [Stock Market Prediction].

Welcome to [Stock Market Prediction], your trusted source for comprehensive stock market analysis, news, and insights. Whether you're a seasoned investor or just starting your journey in the world of finance, we're here to provide you with the tools and information you need to make informed decisions and navigate the complexities of the stock market.

Our Mission:

At [Stock Market Prediction], our mission is to empower individuals with the knowledge and resources to achieve their financial goals. We believe in democratizing access to financial information and fostering a community of informed investors who can confidently navigate the stock market.

What We Offer:

Market Analysis: Stay ahead of market trends with our in-depth analysis and expert commentary on key sectors, stocks, and economic indicators.

News and Updates: Get real-time updates on market news, earnings reports, and major events affecting the stock market.

Research Tools: Access powerful research tools and data analytics to conduct thorough research and identify investment opportunities.

Educational Resources: Learn the fundamentals of investing, trading strategies, and financial planning through our comprehensive educational resources and tutorials.

Community Engagement: Join our vibrant community of investors to share ideas, ask questions, and collaborate with like-minded individuals.

Meet Our Team:

Our team of experienced analysts, researchers, and financial experts is dedicated to providing you with timely and insightful content to help you succeed in the stock market. Get to know the faces behind [Website Name] and learn more about their expertise and passion for finance.

Our History:

Founded in [Year], [Stock Market Prediction] has quickly become a trusted destination for investors seeking reliable market insights and analysis. Over the years, we've grown our platform and expanded our offerings to better serve our growing community of users.

Our Values:

Integrity: We are committed to upholding the highest standards of integrity and ethics in everything we do.

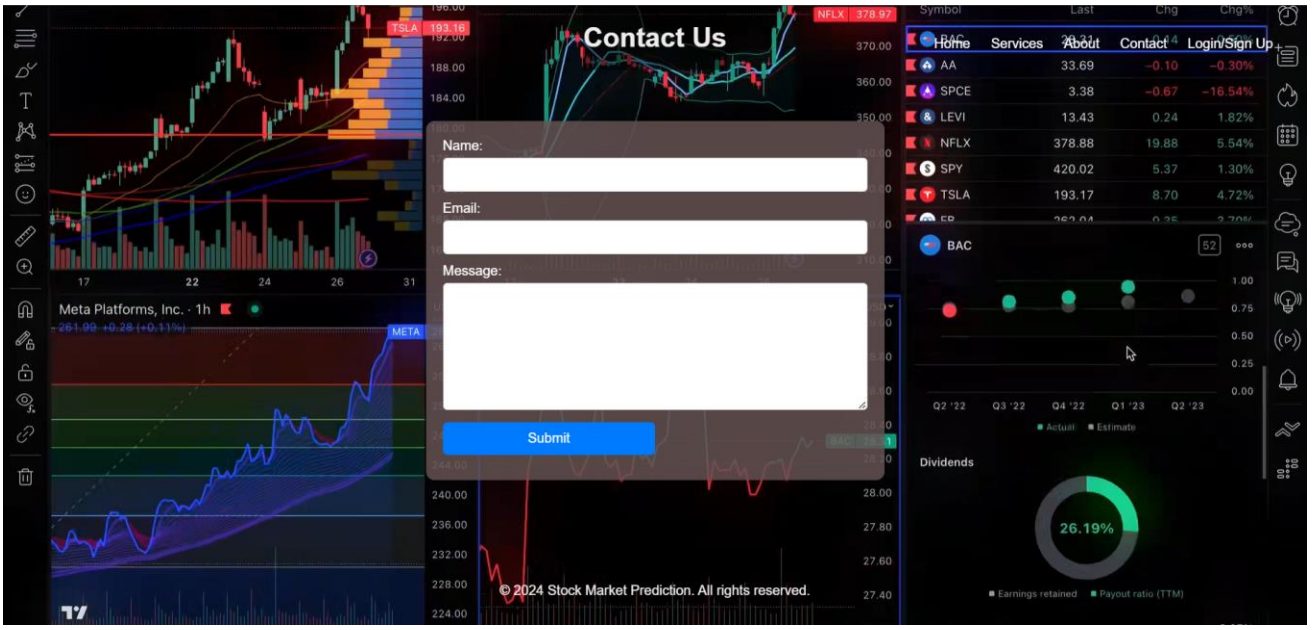
Accuracy: We strive to deliver accurate and reliable information to our users, ensuring they can make informed decisions with confidence.

Transparency: We believe in transparency and openness, providing clear and honest communication with our users at all times.

```
def show_about():
    st.subheader("About")
    display_lottie_animation("about")
    st.write("About page content goes here.")
    st.subheader("Welcome to [Stock Market Prediction].")
    st.write("Welcome to [Stock Market Prediction], your trusted source for comprehensive stock market analysis, news, and insights. Whether you're a seasoned investor or just starting your journey in the world of finance, we're here to provide you with the tools and information you need to make informed decisions and navigate the complexities of the stock market.")
    st.subheader("Our Mission:")
    st.write("At [Stock Market Prediction], our mission is to empower individuals with the knowledge and resources to achieve their financial goals. We believe in democratizing access to financial information and fostering a community of informed investors who can confidently navigate the stock market.")
    st.subheader("What We Offer:")
    st.write("Market Analysis: Stay ahead of market trends with our in-depth analysis and expert commentary on key sectors, stocks, and economic indicators.")
    st.write("News and Updates: Get real-time updates on market news, earnings reports, and major events affecting the stock market.")
    st.write("Research Tools: Access powerful research tools and data analytics to conduct thorough research and identify investment opportunities.")
    st.write("Educational Resources: Learn the fundamentals of investing, trading strategies, and financial planning through our comprehensive educational resources and tutorials.")
    st.write("Community Engagement: Join our vibrant community of investors to share ideas, ask questions, and collaborate with like-minded individuals.")
```

```
    st.subheader("Meet Our Team:")
    st.write("Our team of experienced analysts, researchers, and financial experts is dedicated to providing you with timely and insightful content to help you succeed in the stock market. Get to know the faces behind [Website Name] and learn more about their expertise and passion for finance.")
    st.subheader("Our History:")
    st.write("Founded in [Year], [Stock Market Prediction] has quickly become a trusted destination for investors seeking reliable market insights and analysis. Over the years, we've grown our platform and expanded our offerings to better serve our growing community of users.")
    st.subheader("Our Values:")
    st.write("Integrity: We are committed to upholding the highest standards of integrity and ethics in everything we do.")
    st.write("Accuracy: We strive to deliver accurate and reliable information to our users, ensuring they can make informed decisions with confidence.")
    st.write("Transparency: We believe in transparency and openness, providing clear and honest communication with our users at all times.")
```

Contact.html:-



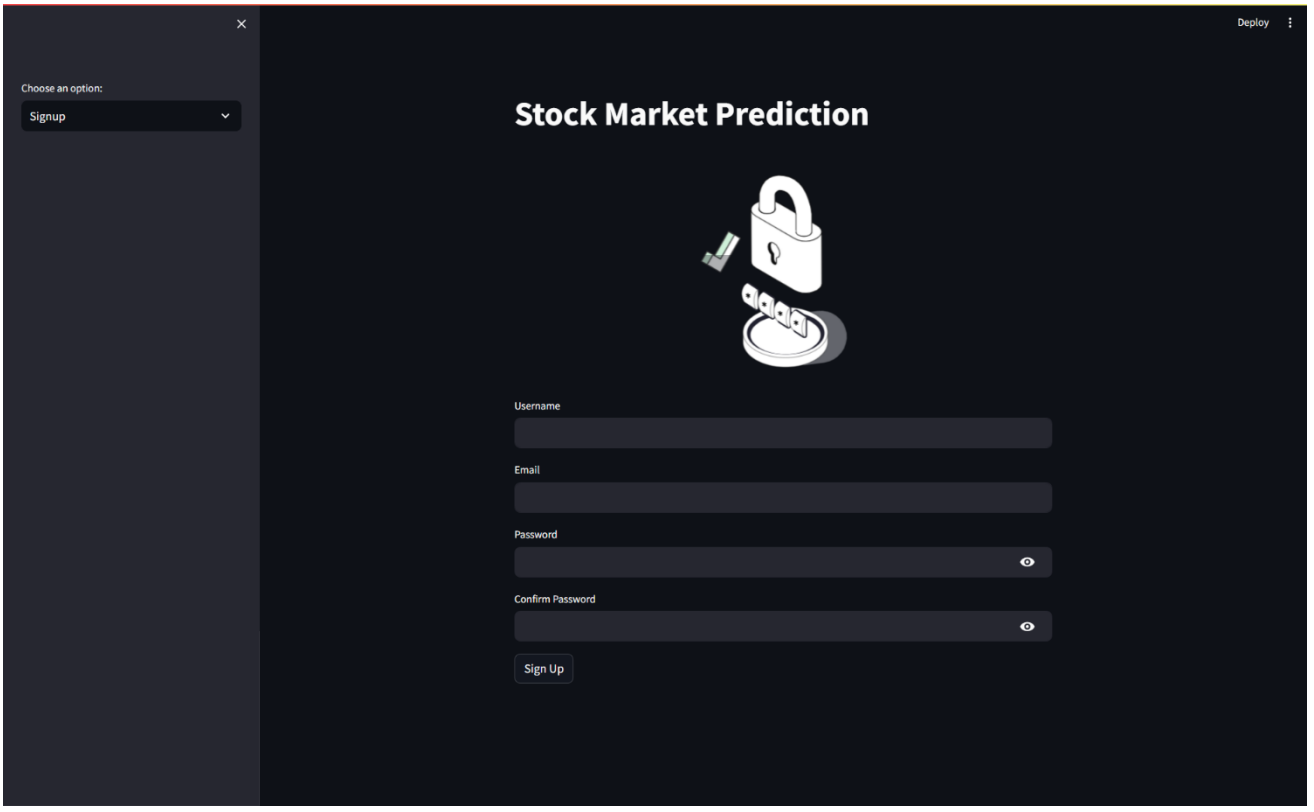

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Contact Page</title>
    <link rel="icon" href="logo-Photoroom.png-Photoroom.png" type="image/x-icon">
    <link rel="stylesheet" href="styles.css">

    <!-- Include Lottie library -->
    <script src="https://cdnjs.cloudflare.com/ajax/libs/bodymovin/5.7.5/lottie.min.js"></script>
    <style>
        body {
            margin: 0;
            padding: 0;
            font-family: Arial, sans-serif;
            display: flex;
            flex-direction: column;
            justify-content: flex-start;
            align-items: center;
            min-height: 100vh; /* Ensure the page fills the viewport */
        }

        main {
            display: flex;
            flex-direction: column;
```

```
    justify-content: flex-start;
    align-items: center;
    width: 100%;
    max-width: 800px; /* Adjust as needed */
    padding: 20px;
    box-sizing: border-box;
    margin-top: 20px; /* Adjust to shift the form towards the top */
}

#video-background {
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    object-fit: cover;
    z-index: -1; /* Ensure the video is behind other elements */
}

form {
    width: 100%; /* Adjust width as needed */
    background: rgba(102, 83, 83, 0.842);
    padding: 20px;
    border-radius: 10px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    color: white;
    position: relative;
    z-index: 1; /* Ensure the form is above the video */
    max-width: 500px; /* Maximum width to maintain readability */
}

form label {
    display: block;
    margin-bottom: 5px;
}

form input[type="text"],
form input[type="email"],
form textarea {
    width: 100%;
    padding: 10px;
    margin-bottom: 10px;
    border: 1px solid #ccc;
    border-radius: 5px;
```

```
            box-sizing: border-box;
            font-size: 16px;
            color: #333;
        }

        form textarea {
            height: 150px; /* Adjust textarea height as needed */
        }

        form input[type="submit"] {
            background-color: #007bff;
            color: white;
            border: none;
            padding: 10px 20px;
            cursor: pointer;
            border-radius: 5px;
            font-size: 16px;
            transition: background-color 0.3s;
        }

        form input[type="submit"]:hover {
            background-color: #045cb9;
        }

    </style>
</head>
<body>
    <video id="video-background" autoplay muted loop>
        <source src="chart-big.hvc1.6af4110d38611a03c3a4.mp4" type="video/mp4">
        Your browser does not support the video tag.
    </video>

    <header>
        <h1>Contact Us</h1>
        <nav>
            <ul>
                <li><a href="./stock.html">Home</a></li>
                <li><a href="./services.html">Services</a></li>
                <li><a href="./about.html">About</a></li>
                <li><a href="./contact.html">Contact</a></li>
                <li><a href="http://localhost:8501/">Login/Sign Up</a></li>
            </ul>
        </nav>
    </header>
```

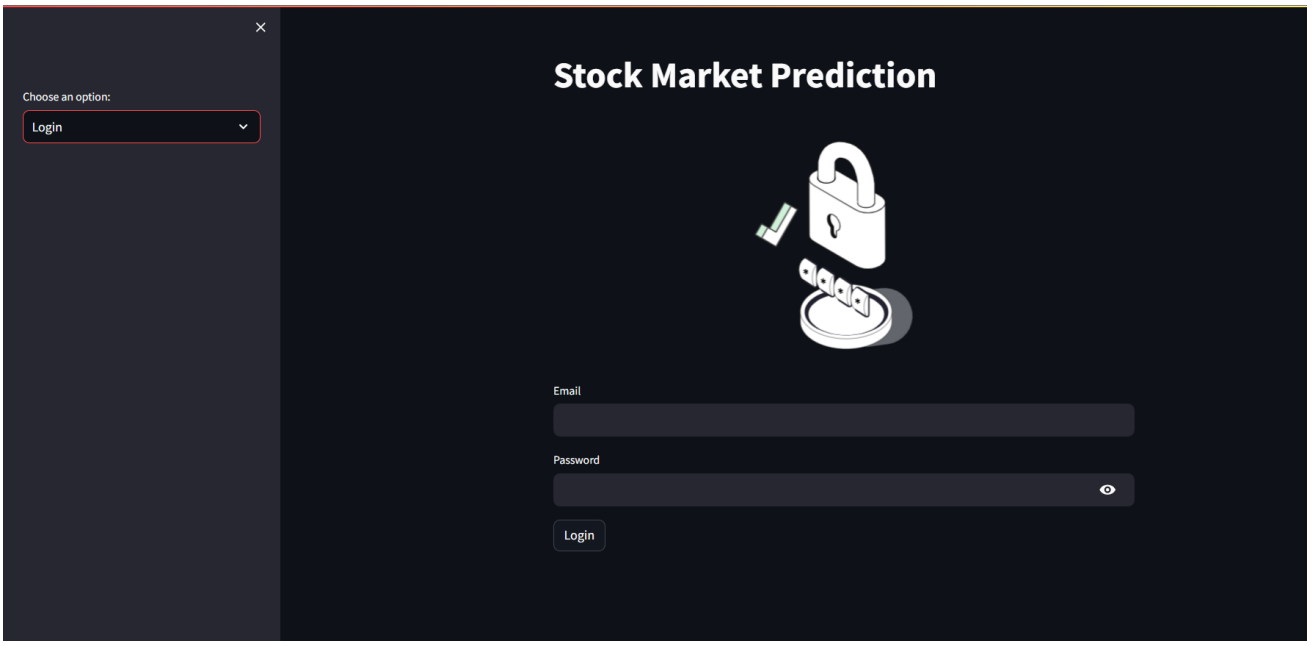
```
    <main>
        <form id="contactForm" action="https://api.web3forms.com/submit" method="POST">
            <input type="hidden" name="access_key" value="d32d1c79-8abd-46f9-bbf7-
88cfa8ec64dc">
            <label for="name">Name:</label>
            <input type="text" id="name" name="name" required>

            <label for="email">Email:</label>
            <input type="email" id="email" name="email" required>

            <label for="message">Message:</label>
            <textarea id="message" name="message" required></textarea>

            <input type="submit" value="Submit">
        </form>
    </main>

    <footer>
        <p>&copy; 2024 Stock Market Prediction. All rights reserved.</p>
    </footer>

    <script>
        // Add event listener to the form submission
        document.getElementById('contactForm').addEventListener('submit', function() {
            // Reload the page after form submission
            location.reload();
        });
    </script>
</body>
</html>
```

Signup.py:-



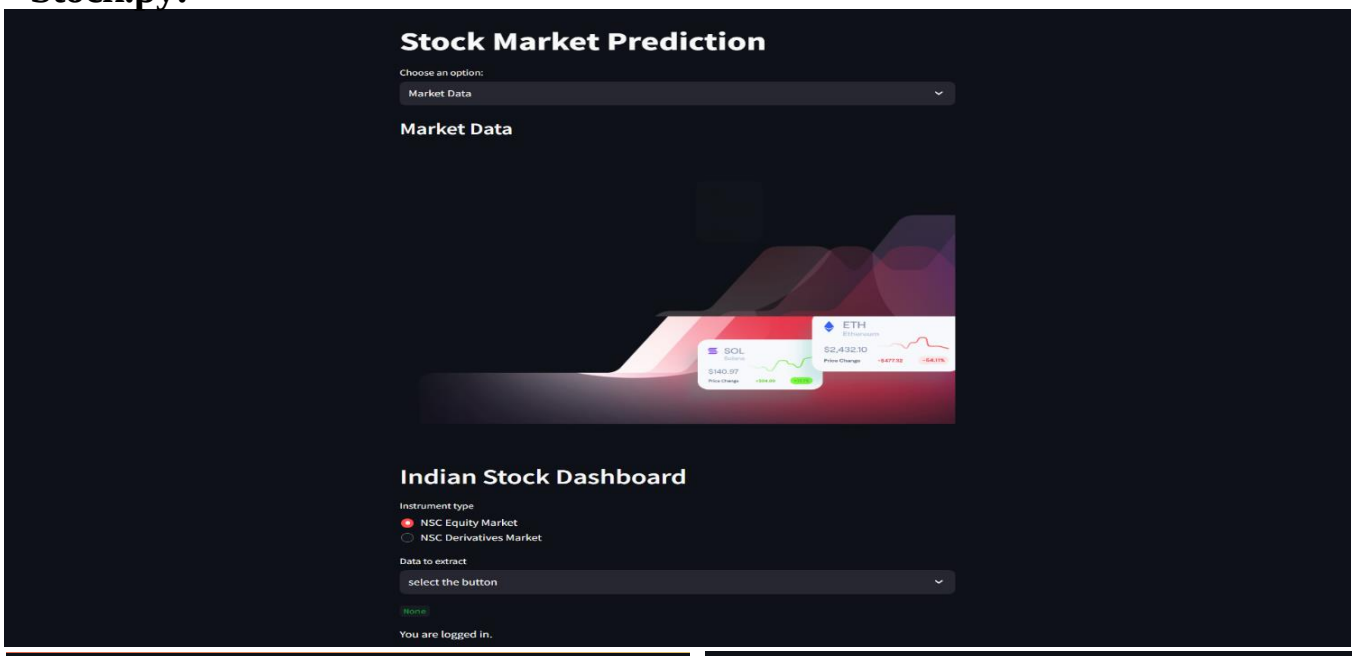

```
elif option == "Signup":
    # Add signup functionality here
    username = st.text_input("Username")
    email = st.text_input("Email")
    password = st.text_input("Password", type="password")
    confirm_password = st.text_input("Confirm Password", type="password")

    if st.button("Sign Up"):
        if password == confirm_password:
            if 4 <= len(password) <= 8:
                # Create the user in the databas
                user_id, error_message = create_user(conn, username, email, password)
                if user_id:
                    st.success("User created successfully!")
                else:
                    st.error(error_message)
            else:
                st.error("Password length must be between 4 and 8 characters.")
        else:
            st.error("Passwords do not match!")
```

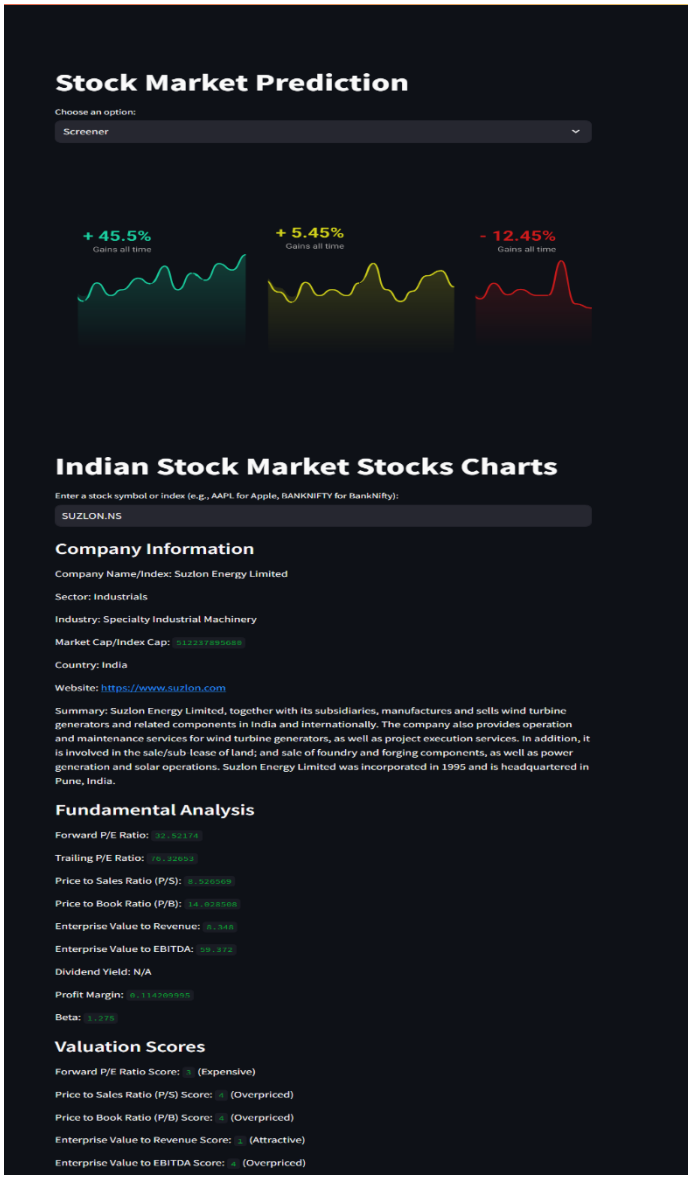
Login.py:-

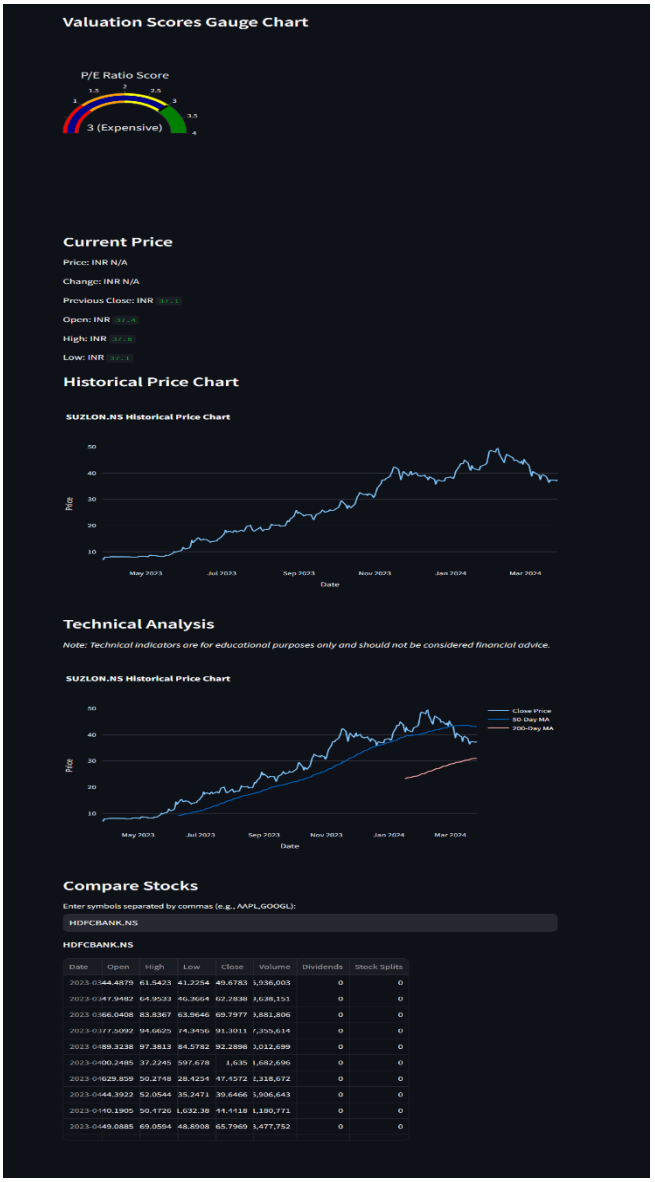



```
elif option == "Login":
    # Add login functionality here
    email = st.text_input("Email")
    password = st.text_input("Password", type="password")

    if st.button("Login"):
        # Check if the user exists in the database
        user = check_user(conn, email, password)
        if user:
            st.session_state.logged_in = True
            st.success(f"Welcome, {user[1]}!")
            # Redirect to the home page
            st.experimental_rerun()
        else:
            st.error("Invalid email or password.")

    pass
```

Stock.py:-


Stock Market Prediction
Choose an option:
Market Data
Market Data
ETH
Ethereum
$2,432.10
SOL
Solana
$140.97
Indian Stock Dashboard
Instrument type
NSC Equity Market
NSC Derivatives Market
Data to extract
select the button
None
You are logged in.


Stock Market Prediction
Choose an option:
Screener
+ 45.5%
Gains all time
+ 5.45%
Gains all time
- 12.45%
Gains all time
Indian Stock Market Stocks Charts
Enter a stock symbol or index (e.g., AAPL for Apple, BANKNIFTY for BankNifty):
SUZLON.NS
Company Information
Company Name/Index: Suzlon Energy Limited
Sector: Industrials
Industry: Specialty Industrial Machinery
Country: India
Website: https://www.suzlon.com
Summary: Suzlon Energy Limited, together with its subsidiaries, manufactures and sells wind turbine generators and related components in India and internationally. The company also provides operation and maintenance services for wind turbine generators, as well as project execution services. In addition, it is involved in the sale/sub lease of land; and sale of foundry and forging components, as well as power generation and solar operations. Suzlon Energy Limited was incorporated in 1995 and is headquartered in Pune, India.
Fundamental Analysis
Forward P/E Ratio:
Trailing P/E Ratio:
Price to Sales Ratio (P/S):
Price to Book Ratio (P/B):
Enterprise Value to Revenue:
Enterprise Value to EBITDA:
Dividend Yield: N/A
Profit Margin:
Beta:
Valuation Scores
Forward P/E Ratio Score: (Expensive)
Price to Sales Ratio (P/S) Score: (Overpriced)
Price to Book Ratio (P/B) Score: (Overpriced)
Enterprise Value to Revenue Score: (Attractive)
Enterprise Value to EBITDA Score: (Overpriced)


Valuation Scores Gauge Chart
P/E Ratio Score
3 (Expensive)
Current Price
Price: INR N/A
Change: INR N/A
Previous Close: INR
Open: INR
High: INR
Low: INR
Historical Price Chart
SUZLON.NS Historical Price Chart
Price
Date
Technical Analysis
Note: Technical indicators are for educational purposes only and should not be considered financial advice.
SUZLON.NS Historical Price Chart
Close Price
50-Day MA
200-Day MA
Compare Stocks
Enter symbols separated by commas (e.g., AAPL,GOOGL):
HDFCBANK.NS
HDFCBANK.NS
Date
Open
High
Low
Close
Volume
Dividends
Stock Splits

Deploy

Enter Stock Ticker Symbol

MSFT

Start Date

2023/01/01

End Date

2024/01/01

Fetch Data

Search

Stock Market Prediction

Choose an option:

Tools

Stock News Show

Data fetched successfully!

Date	Open	High	Low	Close	Adj Close	Volume
2023-01-03 00:00:00	243.08	245.75	237.4	239.58	237.036	25,740,000
2023-01-04 00:00:00	232.28	232.87	225.96	229.1	226.6673	50,623,400
2023-01-05 00:00:00	227.2	227.55	221.76	222.31	219.9494	39,585,600
2023-01-06 00:00:00	223	225.76	219.35	224.93	222.5415	43,613,600
2023-01-09 00:00:00	226.45	231.24	226.41	227.12	224.7083	27,369,800

Stock Price Chart

Enter Stock Ticker Symbol

TATAMOTORS

Start Date

2023/01/01

End Date

2024/01/01

Fetch Data

Search

Deploy

Stock Market Prediction

Choose an option:

Tools

Stock News Show

Latest news for TATAMOTORS:

Headline: India Shares Fall After Three-Session Winning Streak

Source: Biztoc.com

Published: 2024-03-05T13:00:16Z

URL: https://biztoc.com/x/c8bca21418e903f0

Headline: Tata Motors stock soars on EV dominance in India, Jaguar rebound

Source: Biztoc.com

Published: 2024-03-16T14:26:08Z

URL: https://biztoc.com/x/3f57662a772a5c77

Headline: Indian Shares Tick Higher, Market Waits for GDP Data

Source: Biztoc.com

Published: 2024-02-29T11:52:25Z

URL: https://biztoc.com/x/389f15c42944a0d9

Headline: Luxury Car Brands You Should Never Buy

Source: Biztoc.com

Published: 2024-02-28T19:16:10Z

URL: https://biztoc.com/x/5820adfc42f06f72

Headline: Tata confirms Somerset will be home to £4bn battery factory

Source: Biztoc.com

Published: 2024-02-28T08:18:06Z

URL: https://biztoc.com/x/61b978d7730d0184

Stock Market Prediction

Choose an option:

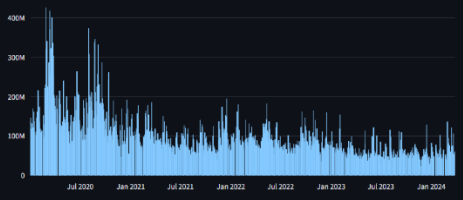
Technical Analysis

Technical Analysis

Stock Prices

Moving Average

Volume

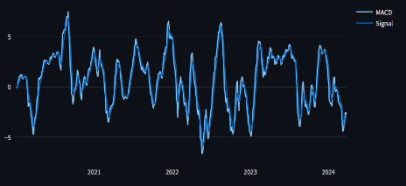


Recent News

- Apple: A Compounding Machine Slowing Down
- Dow Jones Rises After Surprise GDP; Apple Slides On Downgrade
- Can Apple Rise 17%? This Option Trade Offers Such A Return
- Xiaomi Enters Cutthroat EV Race With $29,900 SU7 Series
- Spotify Is Rocking Hard Right Now. Here's Why.
- The Top 2 Dividend Payers From 2023 Were Both Tech Giants
- The Biggest Bear Case for This Top Warren Buffett Stock
- Global smartphone market set to rebound in 2024, report says

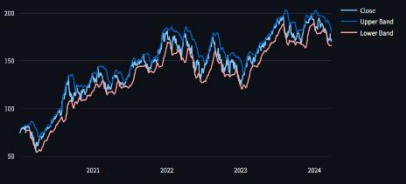
Relative Strength Index (RSI)

Moving Average Convergence Divergence (MACD)

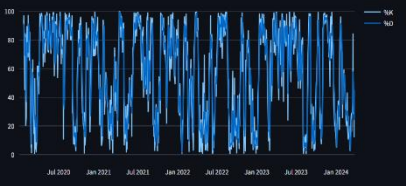


Moving Averages

Bollinger Bands

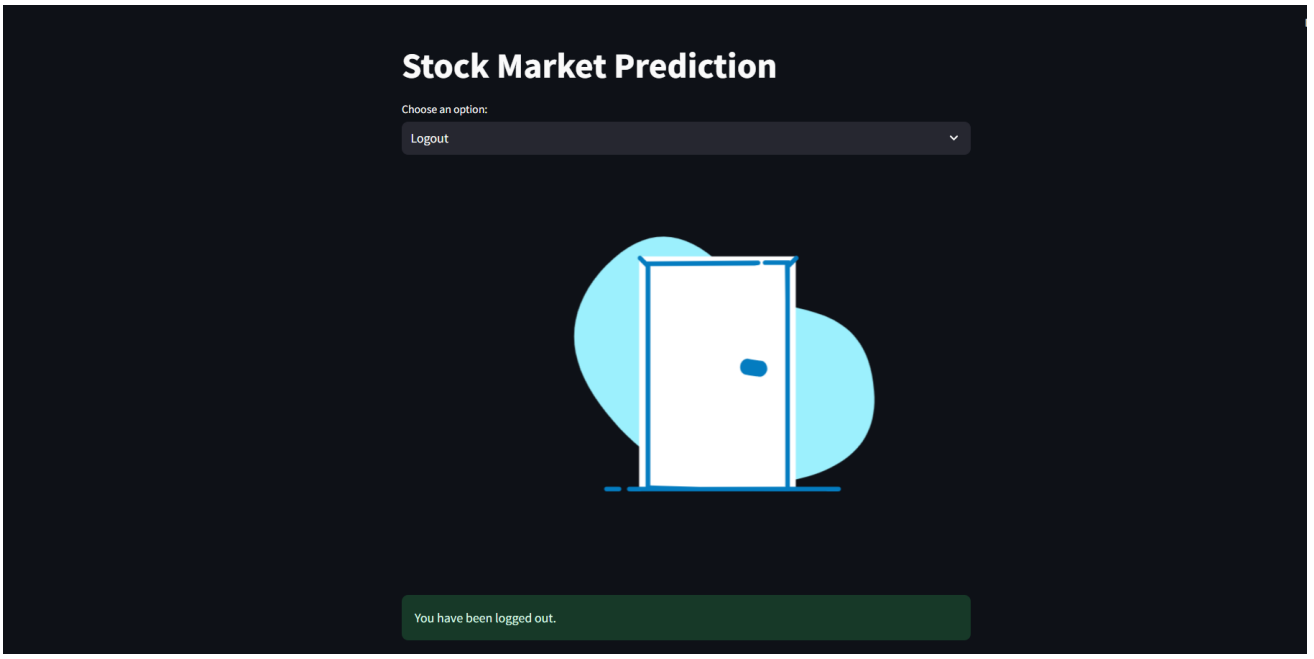


Stochastic Oscillator



Volume Oscillator

Ichimoku Cloud



main.py:-

```
import streamlit as st
import sqlite3
import pandas as pd
from datetime import datetime
import yfinance as yf
import plotly.graph_objs as go
from nselib import capital_market
from nselib import derivatives
from streamlit_lottie import st_lottie
import requests
from textblob import TextBlob

st.set_page_config(page_title='Stock Market Prediction ', page_icon=':chart_with_upwards_trend:')

# Function to create a database connection
def create_connection(db_file):
    conn = None
    try:
        conn = sqlite3.connect(db_file)
    except sqlite3.Error as e:
```

```
        print(e)
    return conn

# Function to create a new user
def create_user(conn, username, email, password):
    cur = conn.cursor()
    # Check if the user already exists
    cur.execute("SELECT * FROM users WHERE email = ?", (email,))
    existing_user = cur.fetchone()
    if existing_user:
        return None, "User with this email already exists. Please use a different email."
    else:
        try:
            sql = ''' INSERT INTO users(username,email,password)
                      VALUES(?,?,?) '''
            cur.execute(sql, (username, email, password))
            conn.commit()
            return cur.lastrowid, None
        except sqlite3.Error as e:
            return None, str(e)

# Function to check if a user exists in the database
def check_user(conn, email, password):
    cur = conn.cursor()
    cur.execute("SELECT * FROM users WHERE email = ? AND password = ?", (email, password))
    user = cur.fetchone()
    return user

lottie_urls = {
    "home": "https://lottie.host/0ef46eab-d90a-42e0-8709-28c337699dc1/XMqrZCglQR.json",
    "screener": "https://lottie.host/2fe89f7a-68cb-49d7-a699-e2e2d88df54b/IVVFddwe7Z.json",
    "tools": "https://lottie.host/cd46ee9c-a82b-48ae-af7d-a80b0658a7ec/jJvr9GFylg.json",
    "about": "https://lottie.host/e5797a37-65c0-44df-8d92-38f72eee4fd9/ogHow1IeLu.json",
    "logout": "https://lottie.host/1c1aa709-2ed7-4fec-b106-012053026d9f/75rl8Nm1rw.json",
```

```
    # Add more pages and their respective Lottie animation URLs here
}

# Function to display the Lottie animation for a specific page
def display_lottie_animation(page_key):
    animation_url = lottie_urls.get(page_key)
    if animation_url:
        response = requests.get(animation_url)
        animation_json = response.json()
        st_lottie(animation_json, speed=1, width=800, height=500,
key=f"lottie_{page_key}_animation")

# Define functions for each page
def show_home():
    st.subheader("Market Data")
    display_lottie_animation("home")
    def fetch_equity_market_data(data_info):
            if data_info in ['equity_list', 'fno_equity_list', 'market_watch_all_indices',
'nifty50_equity_list']:
                return getattr(capital_market, data_info)()
            elif data_info in ['bhav_copy_equities', 'bhav_copy_with_delivery']:
                date = st.sidebar.text_input('Date', '22-12-23')
                parsed_date = datetime.strptime(date, '%d-%m-%y')
                formatted_date = parsed_date.strftime('%d-%m-%Y')
                return getattr(capital_market, data_info)(formatted_date)
            elif data_info in ['block_deals_data', 'bulk_deal_data', 'india_vix_data', 'short_selling_data']:
                period_ = st.sidebar.text_input('Period', '1M')
                return getattr(capital_market, data_info)(period=period_)
    def fetch_derivatives_market_data(data_info):
            if data_info in ['expiry_dates_future', 'expiry_dates_option_index']:
                return getattr(derivatives, data_info)()
            elif data_info in ['fii_derivatives_statistics', 'fno_bhav_copy', 'participant_wise_open_interest',
'participant_wise_trading_volume']:
                date = st.sidebar.text_input('Date', '22-12-23')
```

```
            parsed_date = datetime.strptime(date, '%d-%m-%y')
            formatted_date = parsed_date.strftime('%d-%m-%Y')
            return getattr(derivatives, data_info)(formatted_date)
        elif data_info == 'future_price_volume_data':
            ticker = st.sidebar.text_input('Ticker', 'SBIN')
            type_ = st.sidebar.text_input('Tnstrument Type', 'FUTSTK')
            period_ = st.sidebar.text_input('Period', '1M')
            return derivatives.future_price_volume_data(ticker, type_, period=period_)
        elif data_info == 'option_price_volume_data':
            ticker = st.sidebar.text_input('Ticker', 'BANKNIFTY')
            type_ = st.sidebar.text_input('Tnstrument Type', 'OPTIDX')
            period_ = st.sidebar.text_input('Period', '1M')
            return derivatives.option_price_volume_data(ticker, type_, period=period_)
        elif data_info == 'nse_live_option_chain':
            ticker = st.sidebar.text_input('Ticker', 'BANKNIFTY')
            expiry_data = st.sidebar.text_input('Expiry Data', '28-12-2023')
            return derivatives.nse_live_option_chain(ticker, expiry_date=expiry_data)


    def main():
        st.header('Indian Stock Dashboard')
        instrument = st.radio('Instrument type', options=('NSC Equity Market', 'NSC Derivatives
Market'))
        if instrument == 'NSC Equity Market':
            data_info = st.selectbox('Data to extract', options=('select the button' ,'bhav_copy_equities',
'bhav_copy_with_delivery',
                                                 'equity_list', 'fno_equity_list',
                                                 'market_watch_all_indices', 'nifty50_equity_list',
                                                 'block_deals_data', 'bulk_deal_data',
                                                 'india_vix_data', 'short_selling_data',
                                                 ))
            data = fetch_equity_market_data(data_info)

        elif instrument == 'NSC Derivatives Market':
            data_info = st.selectbox('Data to extract', options=('select the button' ,'expiry_dates_future',
```

```
'expiry_dates_option_index',
                                                'fno_bhav_copy', 'future_price_volume_data',
                                                 'nse_live_option_chain', 'option_price_volume_data',
                                                 'participant_wise_open_interest',
                                                 'participant_wise_trading_volume'))
            data = fetch_derivatives_market_data(data_info)
        st.write(data)
    if __name__ == "__main__":
       main()



    if st.session_state.logged_in:
          st.write("You are logged in.")
    else:
          st.write("You are not logged in.")

def calculate_valuation_scores(info):
    # Define scoring ranges (you can adjust these as needed)
    pe_range = [(0, 15), (15, 25), (25, 35), (35, float('inf'))]  # P/E ratio
    ps_range = [(0, 1), (1, 2), (2, 3), (3, float('inf'))]  # P/S ratio
    pb_range = [(0, 1), (1, 2), (2, 3), (3, float('inf'))]  # P/B ratio
    ev_to_rev_range = [(0, 10), (10, 20), (20, 30), (30, float('inf'))]  # Enterprise Value to Revenue
    ev_to_ebitda_range = [(0, 10), (10, 15), (15, 20), (20, float('inf'))]  # Enterprise Value to EBITDA

    # Get fundamental metrics
    pe_ratio = info.get('forwardPE', 0)
    ps_ratio = info.get('priceToSalesTrailing12Months', 0)
    pb_ratio = info.get('priceToBook', 0)
    ev_to_rev = info.get('enterpriseToRevenue', 0)
    ev_to_ebitda = info.get('enterpriseToEbitda', 0)

    # Calculate scores and labels
    pe_score, pe_label = calculate_score_and_label(pe_ratio, pe_range)
    ps_score, ps_label = calculate_score_and_label(ps_ratio, ps_range)
```

```
    pb_score, pb_label = calculate_score_and_label(pb_ratio, pb_range)
    ev_to_rev_score, ev_to_rev_label = calculate_score_and_label(ev_to_rev, ev_to_rev_range)
    ev_to_ebitda_score, ev_to_ebitda_label = calculate_score_and_label(ev_to_ebitda,
ev_to_ebitda_range)

    return pe_score, pe_label, ps_score, ps_label, pb_score, pb_label, ev_to_rev_score,
ev_to_rev_label, ev_to_ebitda_score, ev_to_ebitda_label

# Function to calculate score and label based on value and ranges
def calculate_score_and_label(value, ranges):
    for i, (lower, upper) in enumerate(ranges):
        if lower <= value < upper:
            if i == 0:
                label = "Attractive"
            elif i == len(ranges) - 1:
                label = "Overpriced"
            else:
                label = "Expensive"
            return i + 1, label  # Scores start from 1
    return len(ranges), "Overpriced"  # If value exceeds last range, return the maximum score

def show_screener():
    display_lottie_animation("screener")
    st.title('Indian Stock Market Stocks Charts')
    symbol = st.text_input("Enter a stock symbol or index (e.g., AAPL for Apple, BANKNIFTY for
BankNifty):")

    if symbol:
        try:
            if symbol.upper() == "BANKNIFTY":
                symbol = "^NSEBANK"  # Yahoo Finance symbol for BankNifty
            stock = yf.Ticker(symbol)
            info = stock.info
```

```
        st.subheader("Company Information" if symbol.upper() != "^NSEBANK" else "Index Information")
        st.write("Company Name/Index:", info.get('longName', 'N/A'))
        st.write("Sector:", info.get('sector', 'N/A'))
        st.write("Industry:", info.get('industry', 'N/A'))
        st.write("Market Cap/Index Cap:", info.get('marketCap', 'N/A'))
        st.write("Country:", info.get('country', 'N/A'))
        st.write("Website:", info.get('website', 'N/A'))
        st.write("Summary:", info.get('longBusinessSummary', 'N/A'))

        if symbol.upper() != "^NSEBANK":
            st.subheader("Fundamental Analysis")
            st.write("Forward P/E Ratio:", info.get('forwardPE', 'N/A'))
            st.write("Trailing P/E Ratio:", info.get('trailingPE', 'N/A'))
            st.write("Price to Sales Ratio (P/S):", info.get('priceToSalesTrailing12Months', 'N/A'))
            st.write("Price to Book Ratio (P/B):", info.get('priceToBook', 'N/A'))
            st.write("Enterprise Value to Revenue:", info.get('enterpriseToRevenue', 'N/A'))
            st.write("Enterprise Value to EBITDA:", info.get('enterpriseToEbitda', 'N/A'))
            st.write("Dividend Yield:", info.get('dividendYield', 'N/A'))
            st.write("Profit Margin:", info.get('profitMargins', 'N/A'))
            st.write("Beta:", info.get('beta', 'N/A'))

            # Calculate valuation scores and labels
            pe_score, pe_label, ps_score, ps_label, pb_score, pb_label, ev_to_rev_score, ev_to_rev_label, ev_to_ebitda_score, ev_to_ebitda_label = calculate_valuation_scores(info)

            st.subheader("Valuation Scores")
            st.write("Forward P/E Ratio Score:", pe_score, "(" + pe_label + ")")
            st.write("Price to Sales Ratio (P/S) Score:", ps_score, "(" + ps_label + ")")
            st.write("Price to Book Ratio (P/B) Score:", pb_score, "(" + pb_label + ")")
            st.write("Enterprise Value to Revenue Score:", ev_to_rev_score, "(" + ev_to_rev_label + ")")
            st.write("Enterprise Value to EBITDA Score:", ev_to_ebitda_score, "(" + ev_to_ebitda_label + ")")
```

```
    # Display gauge chart for valuation scores
    st.subheader("Valuation Scores Gauge Chart")
    fig = go.Figure()

    # Define gauge chart for each score
    fig.add_trace(go.Indicator(
        mode="gauge+number",
        value=pe_score,
        domain={'x': [0, 0.25], 'y': [0.5, 0.9]},
        title={'text': "P/E Ratio Score"},
        gauge={'axis': {'range': [None, 4]},
               'bar': {'color': "darkblue"},
               'steps': [
                   {'range': [0, 1], 'color': "red"},
                   {'range': [1, 2], 'color': "orange"},
                   {'range': [2, 3], 'color': "yellow"},
                   {'range': [3, 4], 'color': "green"}],
               'threshold': {'line': {'color': "black", 'width': 4}, 'thickness': 0.75, 'value': 3}},
        number={'suffix': " (" + pe_label + ")"}
    ))

    # Repeat the above for other scores (P/S, P/B, EV/Rev,
    fig.update_layout(height=400)
    st.plotly_chart(fig, use_container_width=True)

st.subheader("Current Price")
price = info.get('regularMarketPrice', 'N/A')
change = info.get('regularMarketChange', 'N/A')
prev_close = info.get('regularMarketPreviousClose', 'N/A')
open_price = info.get('regularMarketOpen', 'N/A')
high = info.get('regularMarketDayHigh', 'N/A')
low = info.get('regularMarketDayLow', 'N/A')
st.write("Price:", info['currency'], price)
```

```
        st.write("Change:", info['currency'], change)
        st.write("Previous Close:", info['currency'], prev_close)
        st.write("Open:", info['currency'], open_price)
        st.write("High:", info['currency'], high)
        st.write("Low:", info['currency'], low)

        st.subheader("Historical Price Chart")
        history = stock.history(period="1y")
        fig = go.Figure()
        fig.add_trace(go.Scatter(x=history.index, y=history['Close'], mode='lines', name='Close
Price'))
        fig.update_layout(title=f"{symbol} Historical Price Chart", xaxis_title="Date",
yaxis_title="Price")
        st.plotly_chart(fig, use_container_width=True)

        st.subheader("Technical Analysis")
        st.write("*Note: Technical indicators are for educational purposes only and should not be
considered financial advice.*")

        # Example of adding moving averages (50-day and 200-day) to the chart
        history['MA50'] = history['Close'].rolling(window=50).mean()
        history['MA200'] = history['Close'].rolling(window=200).mean()

        fig.add_trace(go.Scatter(x=history.index, y=history['MA50'], mode='lines', name='50-Day
MA'))
        fig.add_trace(go.Scatter(x=history.index, y=history['MA200'], mode='lines', name='200-Day
MA'))

        st.plotly_chart(fig, use_container_width=True)

        # Add an option to compare multiple stocks
        st.subheader("Compare Stocks")
        stocks_to_compare = st.text_input("Enter symbols separated by commas (e.g.,
AAPL,GOOGL):").upper().split(",")
```

```
            if stocks_to_compare:
                for symbol in stocks_to_compare:
                    stock_data = yf.Ticker(symbol)
                    st.write(f"**{symbol}**")
                    st.write(stock_data.history(period="1y"))
        except ValueError:
            st.error("Invalid symbol. Please enter a valid stock symbol or index.")

def fetch_stock_news(symbol):
    api_key = '478378216e84426193f454f310abadb6'
    url = f'https://newsapi.org/v2/everything?q={symbol}&apiKey={api_key}'
    response = requests.get(url)
    if response.status_code == 200:
        return response.json()['articles']
    else:
        return None

def show_tools():
    st.title('Stock News Show')
    display_lottie_animation("tools")
    # Add tools page content here
    def fetch_stock_data(symbol, start_date, end_date):
            stock_data = yf.download(symbol, start=start_date, end=end_date)
            return stock_data

    def display_stock_chart(stock_data):
              fig = go.Figure()
              fig.add_trace(go.Scatter(x=stock_data.index, y=stock_data['Close'], name='Close Price'))
              fig.layout.update(title='Stock Price Chart', xaxis_title='Date', yaxis_title='Price')
              st.plotly_chart(fig)

    symbol = st.sidebar.text_input("Enter Stock Ticker Symbol", value='AAPL')
    start_date = st.sidebar.date_input('Start Date', value=pd.to_datetime('2023-01-01'))
    end_date = st.sidebar.date_input('End Date', value=pd.to_datetime('2024-01-01'))
```

```
if st.sidebar.button('Fetch Data'):
    try:
        stock_data = fetch_stock_data(symbol, start_date, end_date)
        st.success('Data fetched successfully!')
        st.write(stock_data.head())
        display_stock_chart(stock_data)
    except Exception as e:
        st.error(f'Error fetching data: {str(e)}')
# Embedding TradingView chart with custom width and height
st.components.v1.html(f"""
<!-- TradingView Widget BEGIN -->
<div class="tradingview-widget-container" style="width: 100vw; height: 100vh;">
  <div id="tradingview_chart" style="width: 100%; height: 100%;"></div>
  <script type="text/javascript" src="https://s3.tradingview.com/tv.js"></script>
  <script type="text/javascript">
    new TradingView.widget(
    {{
      "autosize": true,
      "symbol": "{symbol}",
      "interval": "D",
      "timezone": "Etc/UTC",
      "theme": "light",
      "style": "1",
      "locale": "en",
      "toolbar_bg": "#f1f3f6",
      "enable_publishing": false,
      "allow_symbol_change": true,
      "container_id": "tradingview_chart",
      "range": "{start_date} {end_date}"
    }}
    );
  </script>
</div>
```

```
    <!-- TradingView Widget END -->
    """, height=600)

    if st.sidebar.button("Search"):
        if symbol:
            news = fetch_stock_news(symbol)
            if news:
                st.write(f"Latest news for {symbol}:")
                for article in news:
                    st.write("Headline:", article['title'])
                    st.write("Source:", article['source']['name'])
                    st.write("Published:", article['publishedAt'])
                    st.write("URL:", article['url'])
                    st.write("---")
            else:
                st.write("Failed to fetch news. Please try again later.")

def show_technical_analysis():
    st.title('Technical Analysis')
    display_lottie_animation("technical_analysis")

    def get_stock_data(symbol, start_date, end_date, interval):
        stock_data = yf.download(symbol, start=start_date, end=end_date, interval=interval)
        return stock_data
    def display_stock_data(stock_data):
        st.subheader('Stock Prices')
        fig = go.Figure(data=[go.Candlestick(x=stock_data.index,
                                             open=stock_data['Open'],
                                             high=stock_data['High'],
                                             low=stock_data['Low'],
                                             close=stock_data['Close'])])
        st.plotly_chart(fig, use_container_width=True)

        # Moving average
```

```
        st.subheader('Moving Average')
        stock_data['MA50'] = stock_data['Close'].rolling(window=50).mean()
        stock_data['MA200'] = stock_data['Close'].rolling(window=200).mean()
        fig_ma = go.Figure()
        fig_ma.add_trace(go.Scatter(x=stock_data.index, y=stock_data['MA50'], mode='lines',
name='MA50'))
        fig_ma.add_trace(go.Scatter(x=stock_data.index, y=stock_data['MA200'], mode='lines',
name='MA200'))
        st.plotly_chart(fig_ma, use_container_width=True)

        # Volume
        st.subheader('Volume')
        fig_volume = go.Figure()
        fig_volume.add_trace(go.Bar(x=stock_data.index, y=stock_data['Volume']))
        st.plotly_chart(fig_volume, use_container_width=True)

    # Function to fetch recent news
    def get_recent_news(symbol):
        news = yf.Ticker(symbol).news
        return news

    # Function to display recent news
    def display_recent_news(news):
        st.subheader('Recent News')
        if len(news) > 0:
            for item in news:
                st.write(f"- {item['title']}")
        else:
            st.write("No recent news found.")

    # Function to display portfolio tracker
    def portfolio_tracker():
        st.sidebar.title('Portfolio Tracker')
        portfolio = st.sidebar.text_input('Enter Stock Symbol', '').upper()
```

```
    if st.sidebar.button('Add to Portfolio'):
        if 'portfolio' not in st.session_state:
            st.session_state.portfolio = []
        if portfolio not in st.session_state.portfolio:
            st.session_state.portfolio.append(portfolio)

    if 'portfolio' in st.session_state:
        st.sidebar.subheader('Portfolio')
        for item in st.session_state.portfolio:
            st.sidebar.write(item)

# Function to display technical indicators
def display_technical_indicators(stock_data):
    st.sidebar.title('Technical Indicators')
    if st.sidebar.checkbox('Relative Strength Index (RSI)'):
        st.subheader('Relative Strength Index (RSI)')
        delta = stock_data['Close'].diff(1)
        gain = (delta.where(delta > 0, 0)).rolling(window=14).mean()
        loss = (-delta.where(delta < 0, 0)).rolling(window=14).mean()
        RS = gain / loss
        RSI = 100 - (100 / (1 + RS))
        fig_rsi = go.Figure()
        fig_rsi.add_trace(go.Scatter(x=stock_data.index, y=RSI, mode='lines', name='RSI'))
        st.plotly_chart(fig_rsi, use_container_width=True)

    if st.sidebar.checkbox('Moving Average Convergence Divergence (MACD)'):
        st.subheader('Moving Average Convergence Divergence (MACD)')
        exp1 = stock_data['Close'].ewm(span=12, adjust=False).mean()
        exp2 = stock_data['Close'].ewm(span=26, adjust=False).mean()
        macd = exp1 - exp2
        signal = macd.ewm(span=9, adjust=False).mean()
        fig_macd = go.Figure()
        fig_macd.add_trace(go.Scatter(x=stock_data.index, y=macd, mode='lines', name='MACD'))
        fig_macd.add_trace(go.Scatter(x=stock_data.index, y=signal, mode='lines', name='Signal'))
```

```
            st.plotly_chart(fig_macd, use_container_width=True)

        # Moving Averages
        if st.sidebar.checkbox('Moving Averages'):
            st.subheader('Moving Averages')
            stock_data['MA50'] = stock_data['Close'].rolling(window=50).mean()
            stock_data['MA200'] = stock_data['Close'].rolling(window=200).mean()
            fig_ma = go.Figure()
            fig_ma.add_trace(go.Scatter(x=stock_data.index, y=stock_data['MA50'], mode='lines',
name='MA50'))
            fig_ma.add_trace(go.Scatter(x=stock_data.index, y=stock_data['MA200'], mode='lines',
name='MA200'))
            st.plotly_chart(fig_ma, use_container_width=True)

        # Bollinger Bands
        if st.sidebar.checkbox('Bollinger Bands'):
            st.subheader('Bollinger Bands')
            stock_data['MA20'] = stock_data['Close'].rolling(window=20).mean()
            stock_data['20D_STD'] = stock_data['Close'].rolling(window=20).std()
            stock_data['UpperBand'] = stock_data['MA20'] + (stock_data['20D_STD'] * 2)
            stock_data['LowerBand'] = stock_data['MA20'] - (stock_data['20D_STD'] * 2)
            fig_bb = go.Figure()
            fig_bb.add_trace(go.Scatter(x=stock_data.index, y=stock_data['Close'], mode='lines',
name='Close'))
            fig_bb.add_trace(go.Scatter(x=stock_data.index, y=stock_data['UpperBand'], mode='lines',
name='Upper Band'))
            fig_bb.add_trace(go.Scatter(x=stock_data.index, y=stock_data['LowerBand'], mode='lines',
name='Lower Band'))
            st.plotly_chart(fig_bb, use_container_width=True)

        # Stochastic Oscillator
        if st.sidebar.checkbox('Stochastic Oscillator'):
            st.subheader('Stochastic Oscillator')
            period = 14
```

```
            stock_data['LowestLow'] = stock_data['Low'].rolling(window=period).min()
            stock_data['HighestHigh'] = stock_data['High'].rolling(window=period).max()
            stock_data['%K'] = ((stock_data['Close'] - stock_data['LowestLow']) /
(stock_data['HighestHigh'] - stock_data['LowestLow'])) * 100
            stock_data['%D'] = stock_data['%K'].rolling(window=3).mean()
            fig_stochastic = go.Figure()
            fig_stochastic.add_trace(go.Scatter(x=stock_data.index, y=stock_data['%K'], mode='lines',
name='%K'))
            fig_stochastic.add_trace(go.Scatter(x=stock_data.index, y=stock_data['%D'], mode='lines',
name='%D'))
            st.plotly_chart(fig_stochastic, use_container_width=True)

        # Volume Oscillator
        if st.sidebar.checkbox('Volume Oscillator'):
            st.subheader('Volume Oscillator')
            short_window = 12
            long_window = 26
            stock_data['Short_MA_Volume'] =
stock_data['Volume'].rolling(window=short_window).mean()
            stock_data['Long_MA_Volume'] =
stock_data['Volume'].rolling(window=long_window).mean()
            stock_data['Volume_Oscillator'] = stock_data['Short_MA_Volume'] -
stock_data['Long_MA_Volume']
            fig_volume_oscillator = go.Figure()
            fig_volume_oscillator.add_trace(go.Scatter(x=stock_data.index,
y=stock_data['Volume_Oscillator'], mode='lines', name='Volume Oscillator'))
            st.plotly_chart(fig_volume_oscillator, use_container_width=True)

        if st.sidebar.checkbox('Ichimoku Cloud'):
            st.subheader('Ichimoku Cloud')
            conversion_period = 9
            base_period = 26
            lagging_period = 52
            displacement = 26
```

```
        high_prices = stock_data['High']
        low_prices = stock_data['Low']
        tenkan_sen = (high_prices.rolling(window=conversion_period).max() +
low_prices.rolling(window=conversion_period).min()) / 2
        kijun_sen = (high_prices.rolling(window=base_period).max() +
low_prices.rolling(window=base_period).min()) / 2
        senkou_span_a = ((tenkan_sen + kijun_sen) / 2).shift(displacement)
        senkou_span_b = ((high_prices.rolling(window=lagging_period).max() +
low_prices.rolling(window=lagging_period).min()) / 2).shift(displacement)
        chikou_span = stock_data['Close'].shift(-displacement)

        fig_ichimoku = go.Figure()
        fig_ichimoku.add_trace(go.Scatter(x=stock_data.index, y=senkou_span_a, mode='lines',
name='Senkou Span A', line=dict(color='blue')))
        fig_ichimoku.add_trace(go.Scatter(x=stock_data.index, y=senkou_span_b, mode='lines',
name='Senkou Span B', line=dict(color='red')))
        fig_ichimoku.add_trace(go.Scatter(x=stock_data.index, y=tenkan_sen, mode='lines',
name='Tenkan Sen', line=dict(color='green')))
        fig_ichimoku.add_trace(go.Scatter(x=stock_data.index, y=kijun_sen, mode='lines',
name='Kijun Sen', line=dict(color='orange')))
        fig_ichimoku.add_trace(go.Scatter(x=stock_data.index, y=chikou_span, mode='lines',
name='Chikou Span', line=dict(color='purple')))
        st.plotly_chart(fig_ichimoku, use_container_width=True)

    # Main function
    def main():
        # Sidebar for user input
        st.sidebar.title('User Input')
        symbol = st.sidebar.text_input('Enter Stock Symbol', value='AAPL', max_chars=20).upper()
        start_date = st.sidebar.date_input('Start Date', value=datetime(2020, 1, 1))
        end_date = st.sidebar.date_input('End Date', value=datetime.today())
        interval = st.sidebar.selectbox('Select Interval', ['1d', '1wk', '1mo'])

        # Fetch and display stock data
```

```
        try:
            stock_data = get_stock_data(symbol, start_date, end_date, interval)
            if not stock_data.empty:
                display_stock_data(stock_data)
            else:
                st.error("No data found for the selected symbol.")

            # Fetch and display recent news
            recent_news = get_recent_news(symbol)
            display_recent_news(recent_news)

            # Display portfolio tracker
            portfolio_tracker()

            # Display technical indicators
            display_technical_indicators(stock_data)
        except Exception as e:
            st.error(f"Error: {e}")
    if __name__ == "__main__":
     main()



def show_about():
    st.subheader("About")
    display_lottie_animation("about")
    st.write("About page content goes here.")
    st.subheader("Welcome to [Stock Market Prediction].")
    st.write("Welcome to [Stock Market Prediction], your trusted source for comprehensive stock market analysis, news, and insights. Whether you're a seasoned investor or just starting your journey in the world of finance, we're here to provide you with the tools and information you need to make informed decisions and navigate the complexities of the stock market.")
    st.subheader("Our Mission:")
    st.write("At [Stock Market Prediction], our mission is to empower individuals with the knowledge
```

```
and resources to achieve their financial goals. We believe in democratizing access to financial information and fostering a community of informed investors who can confidently navigate the stock market.")
    st.subheader("What We Offer:")
    st.write("Market Analysis: Stay ahead of market trends with our in-depth analysis and expert commentary on key sectors, stocks, and economic indicators.")
    st.write("News and Updates: Get real-time updates on market news, earnings reports, and major events affecting the stock market.")
    st.write("Research Tools: Access powerful research tools and data analytics to conduct thorough research and identify investment opportunities.")
    st.write("Educational Resources: Learn the fundamentals of investing, trading strategies, and financial planning through our comprehensive educational resources and tutorials.")
    st.write("Community Engagement: Join our vibrant community of investors to share ideas, ask questions, and collaborate with like-minded individuals.")
    st.subheader("Meet Our Team:")
    st.write("Our team of experienced analysts, researchers, and financial experts is dedicated to providing you with timely and insightful content to help you succeed in the stock market. Get to know the faces behind [Website Name] and learn more about their expertise and passion for finance.")
    st.subheader("Our History:")
    st.write("Founded in [Year], [Stock Market Prediction] has quickly become a trusted destination for investors seeking reliable market insights and analysis. Over the years, we've grown our platform and expanded our offerings to better serve our growing community of users.")
    st.subheader("Our Values:")
    st.write("Integrity: We are committed to upholding the highest standards of integrity and ethics in everything we do.")
    st.write("Accuracy: We strive to deliver accurate and reliable information to our users, ensuring they can make informed decisions with confidence.")
    st.write("Transparency: We believe in transparency and openness, providing clear and honest communication with our users at all times.")

def logout():

    # Perform logout actions here
```

```
    display_lottie_animation("logout")
    st.session_state.logged_in = False
    st.success("You have been logged out.")

def main():
    st.title("Stock Market Prediction")
    # Create a connection to the SQLite database
    conn = create_connection("user_db.sqlite")



    # Create a users table if it doesn't exist
    with conn:
        conn.execute('''CREATE TABLE IF NOT EXISTS users (
                        id INTEGER PRIMARY KEY,
                        username TEXT NOT NULL,
                        email TEXT NOT NULL,
                        password TEXT NOT NULL
                    )''')

    # Initialize session state
    if 'logged_in' not in st.session_state:
        st.session_state.logged_in = False
    url = "https://lottie.host/b1bbf192-fb29-4199-9a7a-60bc8b875067/uAPR21HfI6.json"
    response = requests.get(url)
    animation_json = response.json()

    # Display Lottie animation only on the login page
    if st.session_state.logged_in == False:
      st_lottie(animation_json, speed=3, width=700, height=300, key="lottie_animation")

    # Sidebar options based on login status
    if st.session_state.logged_in:
        option = st.selectbox("Choose an option:", ("Market Data","Screener", "Tools", "Technical
```

```
Analysis",  "About", "Logout"))
    else:
        option = st.sidebar.selectbox("Choose an option:", ("Signup","Login"))

    # Check the option selected
    if option == "Market Data":
        show_home()
    elif option == "Screener":
        show_screener()
    elif option == "Tools":
        show_tools()
    elif option == "Technical Analysis":
        show_technical_analysis()
    elif option == "About":
        show_about()

    elif option == "Logout":
        logout()
    elif option == "Login":
        # Add login functionality here
        email = st.text_input("Email")
        password = st.text_input("Password", type="password")

        if st.button("Login"):
            # Check if the user exists in the database
            user = check_user(conn, email, password)
            if user:
                st.session_state.logged_in = True
                st.success(f"Welcome, {user[1]}!")
                # Redirect to the home page
                st.experimental_rerun()
            else:
                st.error("Invalid email or password.")
```

```
        pass
    elif option == "Signup":
        # Add signup functionality here
        username = st.text_input("Username")
        email = st.text_input("Email")
        password = st.text_input("Password", type="password")
        confirm_password = st.text_input("Confirm Password", type="password")

        if st.button("Sign Up"):
            if password == confirm_password:
                if 4 <= len(password) <= 8:
                    # Create the user in the databas
                    user_id, error_message = create_user(conn, username, email, password)
                    if user_id:
                        st.success("User created successfully!")
                    else:
                        st.error(error_message)
                else:
                    st.error("Password length must be between 4 and 8 characters.")
            else:
                st.error("Passwords do not match!")


if __name__ == "__main__":
    main()
```

Code Efficiency:

Code efficiency is a critical aspect of software development that focuses on optimizing the performance and resource utilization of a program. Efficient code not only executes tasks quickly but also minimizes memory usage and maximizes scalability. Achieving code efficiency requires careful design, implementation, and optimization techniques to ensure that the program performs well under various conditions and scales effectively with increasing workload or data volume.

One key aspect of code efficiency is algorithmic efficiency, which involves selecting and implementing algorithms that minimize computational complexity and resource consumption. This includes choosing the most suitable algorithmic approach for a given problem, optimizing data structures for efficient storage and retrieval, and reducing unnecessary computations or redundant operations.

Advantages of Code Optimization -

1. Optimized code has faster execution speed
2. Optimized code utilizes the memory efficiently
3. Optimized code gives better performance

Testing Approach :

A comprehensive testing approach is essential for ensuring the reliability, functionality, and quality of software applications. A robust testing strategy encompasses various testing techniques, methodologies, and tools to detect defects, verify functionality, and validate user requirements throughout the software development lifecycle.

The testing approach typically begins with the creation of a test plan, which outlines the testing objectives, scope, resources, and timelines. It also defines the testing methodologies and techniques to be employed, such as unit testing, integration testing, system testing, and acceptance testing.

There are several types of testing :

1. Unit Testing: This type of testing involves testing individual units or components of the software in isolation to ensure they work correctly. It is typically performed by developers and focuses on validating the smallest units of code, such as functions or methods.

2. Integration Testing: Integration testing verifies the interactions between different units or modules of the software to ensure they work together as expected. It checks for interface compatibility, data flow, and communication between integrated components.

3. System Testing: System testing evaluates the behavior and functionality of the entire software system as a whole. It tests the integrated system against the specified requirements to ensure it meets user expectations and performs as intended in the target environment.

4. Acceptance Testing: Acceptance testing validates that the software meets the acceptance criteria and satisfies user needs. It is typically performed by end-users or stakeholders to determine whether the software is ready for deployment.

5. Regression Testing: Regression testing verifies that recent code changes have not adversely affected existing functionalities. It involves retesting previously tested features to ensure they still work correctly after modifications or enhancements.

6. Functional Testing: Functional testing assesses the functionality of the software by testing its features against the functional requirements. It verifies that the software performs the intended operations and produces the expected outputs.

7. Non-Functional Testing: Non-functional testing evaluates the non-functional aspects of the software, such as performance, usability, reliability, security, and scalability. Examples include performance testing, usability testing, security testing, and reliability testing.

8. User Interface (UI) Testing: UI testing checks the graphical user interface (GUI) of the software to ensure it is user-friendly, visually appealing, and operates correctly. It verifies that users can interact with the software interface effectively and intuitively.

9. Compatibility Testing: Compatibility testing assesses the compatibility of the software with different platforms, devices, browsers, and operating systems. It ensures that the software functions correctly across various environments and configurations.

10. Security Testing: Security testing identifies vulnerabilities, weaknesses, and potential security risks in the software to protect it from unauthorized access, data breaches, and malicious attacks. It includes techniques such as penetration testing, vulnerability scanning, and security code reviews.


Unit Testing
Tun testing
Systing
Unitpesting
UI
testing
Teststing
Non-Funcration
Security testing
Acceptation
testing
Testing

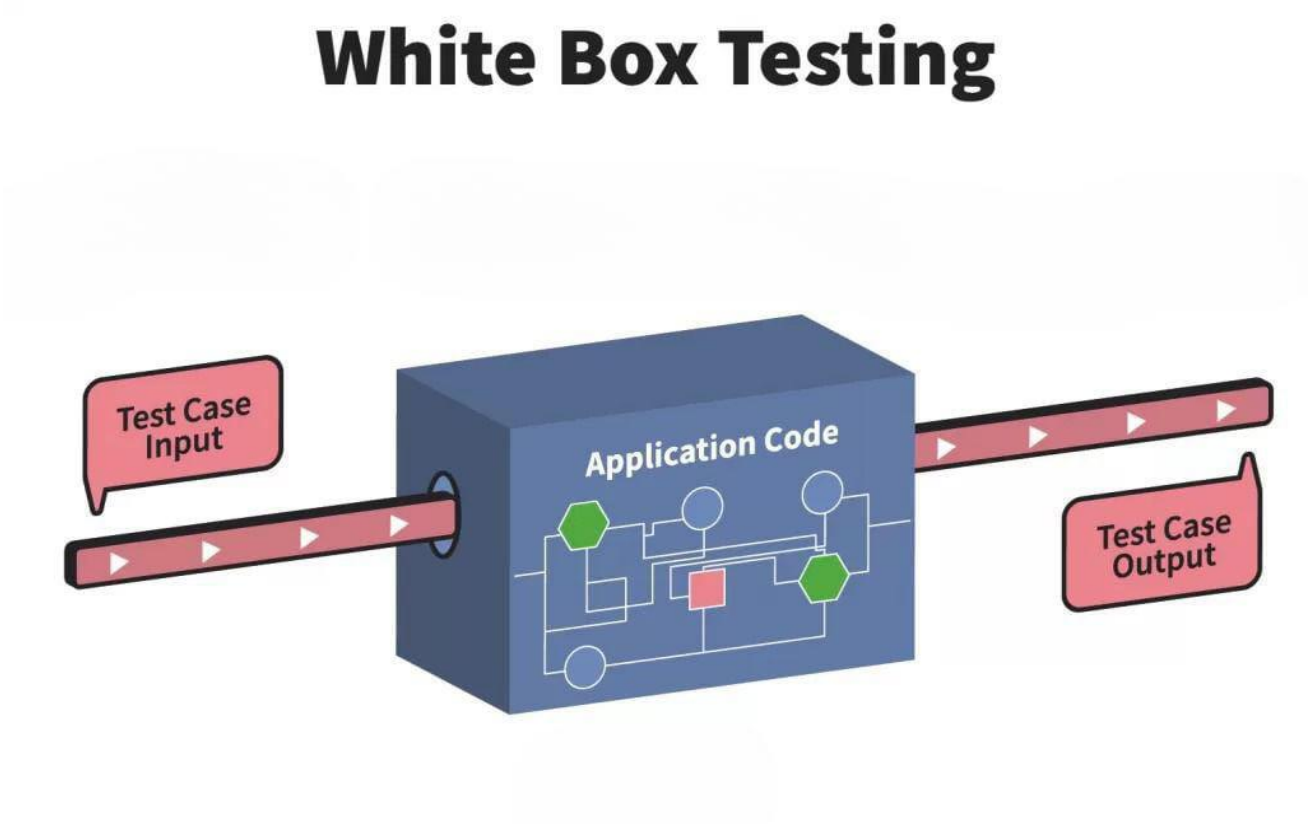
White box testing :

White-box testing, also known as clear-box testing, glass-box testing, or structural testing, is a software testing technique that examines the internal structure and logic of the software application. Unlike black-box testing, which focuses on testing the functionality of the software without knowledge of its internal workings, white-box testing involves inspecting the code, architecture, and design of the software to uncover defects, errors, and vulnerabilities.

In white-box testing, testers have access to the source code and use this knowledge to design test cases that exercise specific paths, conditions, and statements within the code. This allows testers to evaluate the correctness of individual functions, methods, and modules, as well as the overall program flow and control flow.

Process of White Box Testing

Step 1: Identify the feature, component, program to be tested. ...
Step 2: Plot all possible paths in a flowgraph. ...
Step 3: Identify all possible paths from the flowgraph. ...
Step 4: Write Test Cases to cover every single path on the flowgraph. ...
Step 5: Execute, rinse, repeat.



7.2 Levels Of Testing :

1. **Unit Testing:**

   Unit testing is a fundamental software testing technique that focuses on verifying the correctness of individual units or components of a software application. A unit refers to the smallest testable part of a software system, typically a function, method, or procedure.

Test ID	Test scenario	Value	Expected Result	Actual Result	Pass/fail
A1	Splash Screen	True	Working	Forward to next activity	Pass
A2	User Login	Username=user Password = user	Working	Pass on to the Select options.	Pass
A3	User Signup	Username=user (Alphanumeric) Email =user Password=user (Alphanumeric)	Working	Pass on to the Select options.	Pass
A4	Database Connectivity	Data Store	Data To Be Stored	Information Is Stored	Pass

- I have checked each and every unit thoroughly and resolved the errors which arrived during the development of the unit in the project. The login section is divided into admin phase and Employee phase.

  Each having their own characteristics.

2. Integrated Testing :

Integrated testing is a software testing technique that focuses on verifying the interactions and interfaces between different modules or components of a software application.

Test Case ID	Test scenario	Value	Expected Result	Actual Result	Pass/fail
B1	Splash Screen	True	Working	Forward to next activity	Pass
B2	Home button	Entire home screen	Working	Load Successfully	Pass
B3	Services	To explore button get services	Working	Updated Successfully.	Pass
B4	About	Detail of website	Working	Successfully.	Pass
B5	Contact	To from	Working	From submitted successfully	Pass
B6	Login/Sign Up button	User login & signup	Working	Successfully.	Pass

- When all the units are combined and tested together, then the workflow of the project is understood. The login credentials when matched, navigates to the corresponding pages for user.

Beta Testing :

Beta testing is a crucial phase in the software development lifecycle where a pre-release version of the software, known as the beta version, is made available to a selected group of external users or customers for testing and evaluation. This testing phase allows developers to gather feedback from real-world users, identify potential issues, and make necessary improvements before the final release to the wider audience.

Beta testing offers several benefits to software developers and organizations. Firstly, it provides valuable insights into how users interact with the software in real-world scenarios, helping identify usability issues and areas for improvement that may not have been apparent during internal testing. Additionally, beta testing helps increase user engagement and build excitement around the upcoming release, as participants feel involved in the development process and have the opportunity to influence the final product.

Acceptance Testing :

Acceptance testing is the final phase of software testing before the software is released to the end-users or customers. It involves evaluating whether the software meets the requirements and expectations specified by the stakeholders. In simple terms, acceptance testing determines if the software is "accepted" or approved for deployment.

During acceptance testing, users or representatives of the end-users interact with the software to validate that it functions correctly and fulfills their needs. The focus is on assessing whether the software meets the business objectives, user requirements, and quality standards defined during the project's initiation and development phases.

System Testing :

System testing is a critical phase in the software development process where the entire software system is tested as a whole to ensure that it meets the specified requirements and functions correctly in the intended environment. During system testing, the software is evaluated against its functional and non-functional requirements to verify its overall behavior, performance, and reliability.

System testing encompasses various types of testing, including functional testing to validate the functionality of the software, usability testing to assess the user experience, performance testing to evaluate the system's responsiveness and scalability, and security testing to identify potential vulnerabilities. By conducting comprehensive system testing, organizations can ensure that the software meets quality standards and is ready for deployment to end-users.

7.3 Test Cases

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Remark
TC1	Splash Screen					
TC2	Check Signup of User	Enter Email, Username, Password, and Click on Signup	Username = User Email= Akk11@ Password = admin	User should be Signup in	Signup Successfully	Pass
TC3	Check Login of User	Enter Email, Password and Click on Login	Email = Akki Password = Akki11	User should be logged in	Login Successfully	Pass
TC4	SQL Query	Insert, Update, Delete	The data is taken.	Showing data.	Visible data successfully	Pass

COST ESTIMATION

Cost estimate is the approximation of the cost of a program, project, or operation. Accurate cost estimation is crucial for successful project planning and execution. Project cost estimation involves forecasting the financial resources needed to complete a project within a defined scope. It considers direct costs, indirect costs, and other project-related expenses. The goal is to calculate a budget that aligns with the financial commitment required for a successful project. Project managers and estimators use a **cost breakdown structure (CBS)** to identify all costs associated with the project.

The Development Mode:

In the context of COCOMO (Constructive Cost Model), the term "development mode" refers to one of the key factors that influence the estimation of software development effort and cost. COCOMO is a widely used software cost estimation model developed by Barry Boehm, and it considers various factors that affect the effort required to develop a software project.

The development mode in COCOMO refers to the level of familiarity, experience, and maturity of the development team with respect to the project and the technologies involved. It is categorized into three modes:

1. **Organic Mode:**
    - In this mode, the development team has a high level of familiarity with the project's requirements and the technology used.
    - The team has well-defined processes and procedures in place, and there is minimal complexity in the project environment.
    - Organic mode projects are typically smaller in size, and the development team can work relatively independently with minimal external dependencies.

2. **Semi-Detached Mode:**
    - Semi-detached mode represents a moderate level of familiarity and experience of the development team.
    - The project may involve some new technologies or requirements that the team has limited experience with.
    - There may also be moderate complexity in the project environment, such as moderate size or some external dependencies.

- Semi-detached mode projects require a moderate level of coordination and adaptation from the development team.

3. Embedded Mode:

- Embedded mode indicates a low level of familiarity and experience of the development team with the project requirements and technologies.
- The project environment is characterized by high complexity, such as large size, criticality, or significant external dependencies.
- Embedded mode projects typically require extensive coordination, collaboration, and adaptation from the development team to address the challenges and risks involved.

Basic model:

The basic COCOMO equations take the form
Effort Applied (E) = ab(KLOC)bb**[person-months]**
Development Time (D) = cb(Effort Applied)db**[months]**
People Required (P) = Effort Applied / Development time **[count]**

1) **Compute the count-total which will be used to define the complexity of a project.**

Information Domain Values

Measurement Parameter	Count		Simple ○	Average ◉	Complex ○		Total
Number of user inputs	6	X	3	4	6	=	24.00
Number of user outputs	8	X	4	5	7	=	40.00
Number of user inquiries	3	X	3	4	6	=	12.00
Number of files	0	X	7	10	15	=	.0
Number of external interfaces	5	X	5	7	10	=	35.00
Count=Total							111.00

Count Total

2) **Find the complexity adjustment values based on responses to the questions.**

Complexity Weighting Factors
// heading of the second table Rate each factor on a scale of 0 to 5:
(0 = No influence, 1 = Incidental, 2 = Moderate, 3 = Average, 4 = Significant, 5 = Essential):

Question	0	1	2	3	4	5
1. Does the system require reliable backup and recovery?	○	○	○	○	○	◉
2. Are data communications required?	○	○	○	○	○	◉
3. Are there distributed processing functions?	○	○	○	○	◉	○
4. Is performance critical?	○	○	○	○	◉	○
5. Will the system run in an existing, heavily utilized operational environment?	○	○	○	◉	○	○
6. Does the system require on-line data entry?	○	○	○	◉	○	○
7. Does the on-line data entry require the input transaction to be built over multiple screens or operations?	○	○	○	○	◉	○
8. Are the master file updated on-line?	○	○	○	◉	○	○
9. Are the inputs, outputs, files, or inquiries complex?	○	○	○	○	◉	○
10. Is the internal processing complex?	○	○	○	◉	○	○
11. In the code designed to be reusable?	○	○	○	◉	○	○
12. Are conversion and installation included in the design?	○	○	◉	○	○	○
13. Is the system designed for multiple installations in different organizations?	○	○	○	◉	○	○
14. Is the application designed to facilitate change and ease of use by the user?	○	○	○	○	◉	○
Total 50.00						

Show Total of weighting Factor

The Function Points is: Show Function Points 127.65

3) **Select a programming language used in the project.**

Programming Language	LOC/FP (average)	Select
Assembly Language	320	○
C	128	○
COBOL	105	○
Fortran	105	○
Pascal	90	○
Ada	70	○
Object-Oriented Languages	30	○
Fourth Generation Languages (4GLs)	20	○
Code Generators	15	○
Spreadsheets	6	○
Graphical Languages (icons)	4	◉

LOC/F P: Show LOC/FP 510.60

4) **Select complexity of the software project.**

Software Project	a_b	b_b	c_b	d_b	Select
Organic	2.4	1.05	2.5	0.38	◉
Semi-detached	3.0	1.12	2.5	0.35	○
Embedded	3.6	1.20	2.5	0.32	○

Calculate Effort and Duration

Effort (E) = $a_b(KLOC)^{b}_{b}$ = 1.18 Duration (D) = $c_b(E)^{d}_{b}$ = 2.67

Man-month = Unadjusted Function Point (UFP) / 18

= 127 / 18

= 7

Average Programmer is paid Rs. 4000 per

month Total number of programmers: 2

Cost per month = Average Programmer cost * No of programmers

= 4000 * 2

= Rs.8,000 per month

Total cost of Project = Cost per month * man-month

= 8,000 * 7

= Rs. 56,000

RESULT AND DISCUSSION

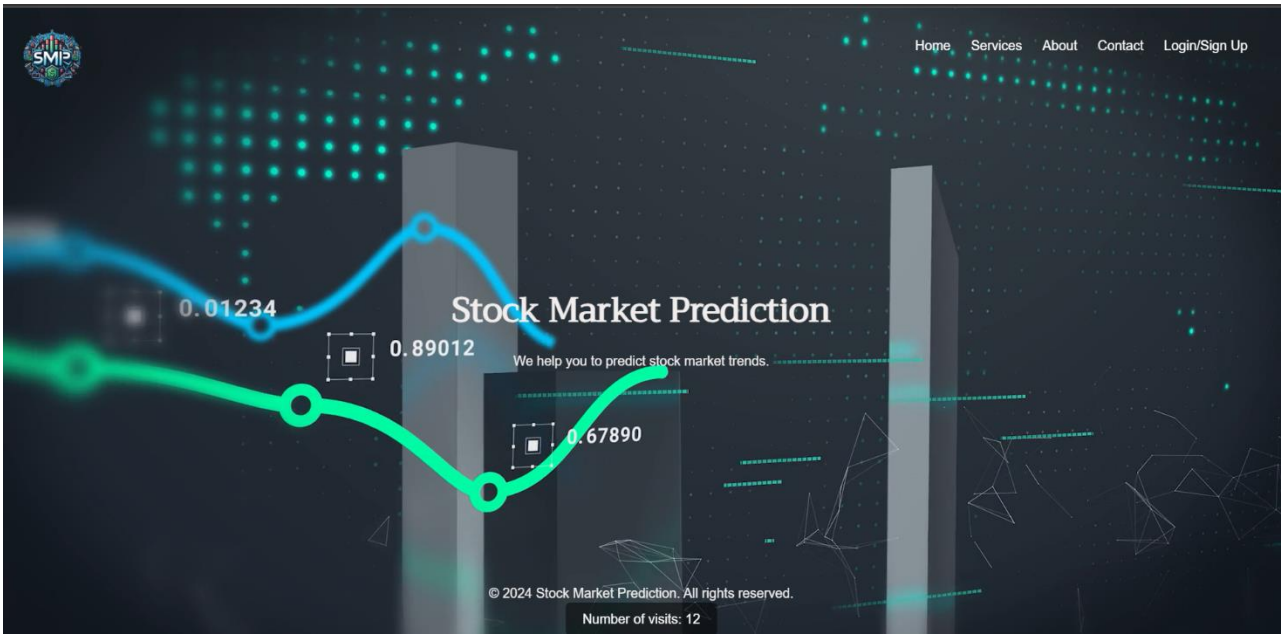
9.1 Test Report

Test cases are detailed scenarios or conditions under which a system application is tested to ensure that it functions correctly and meets its requirements. Each test case typically consists of inputs, expected outputs, and execution conditions. Test cases are designed to verify that the software behaves as expected under various circumstances and to identify any defects or bugs in the system.

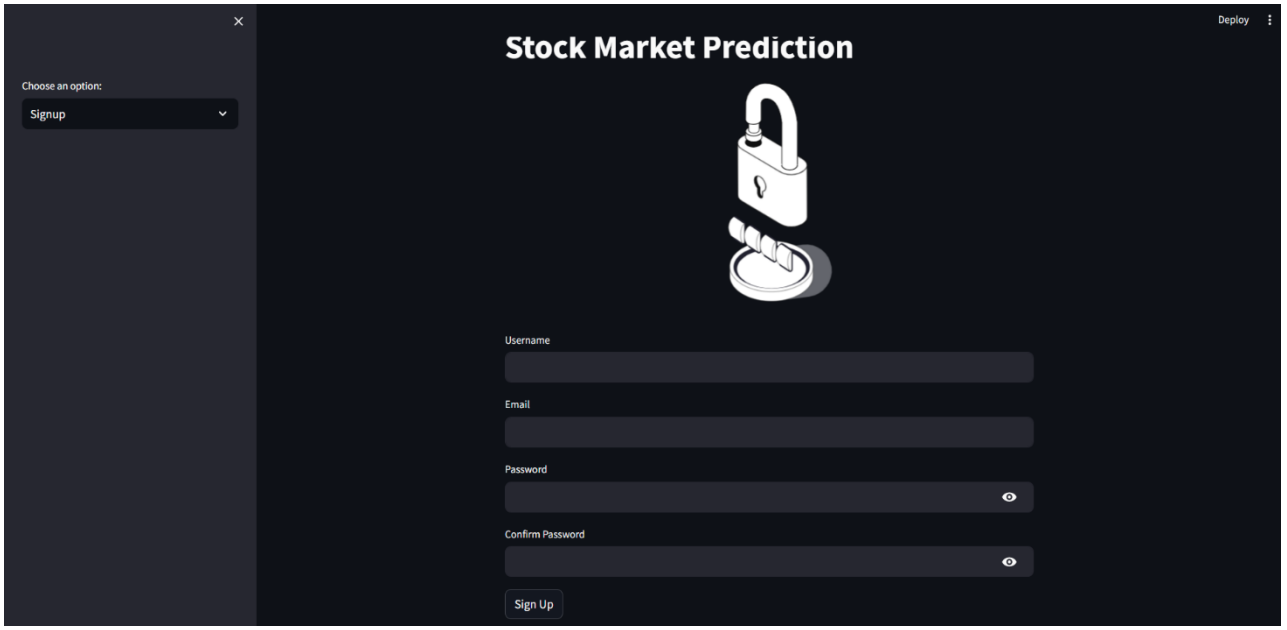
The Test Cases which were performed led to the checking of each and every module in the application. The Splash Screen was tested continuing with the Login of both the admin as well as the Employee activity.

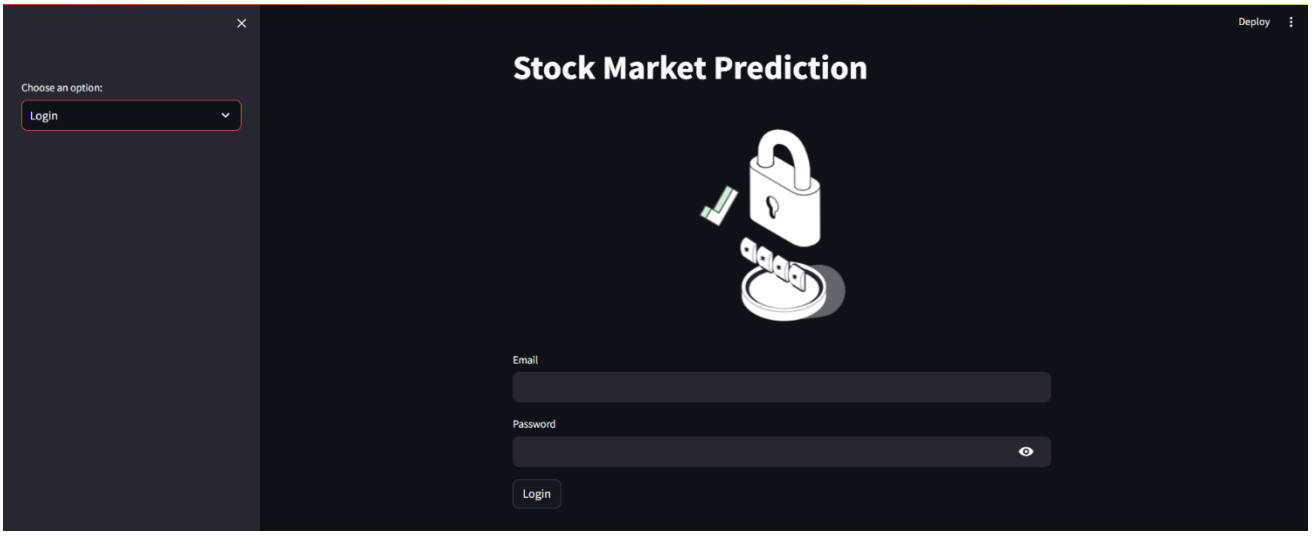
The Database is tested by adding, inserting, updating and deletion of the record. The Data fetching process is also considered during the applications.

A) Splash Screen :

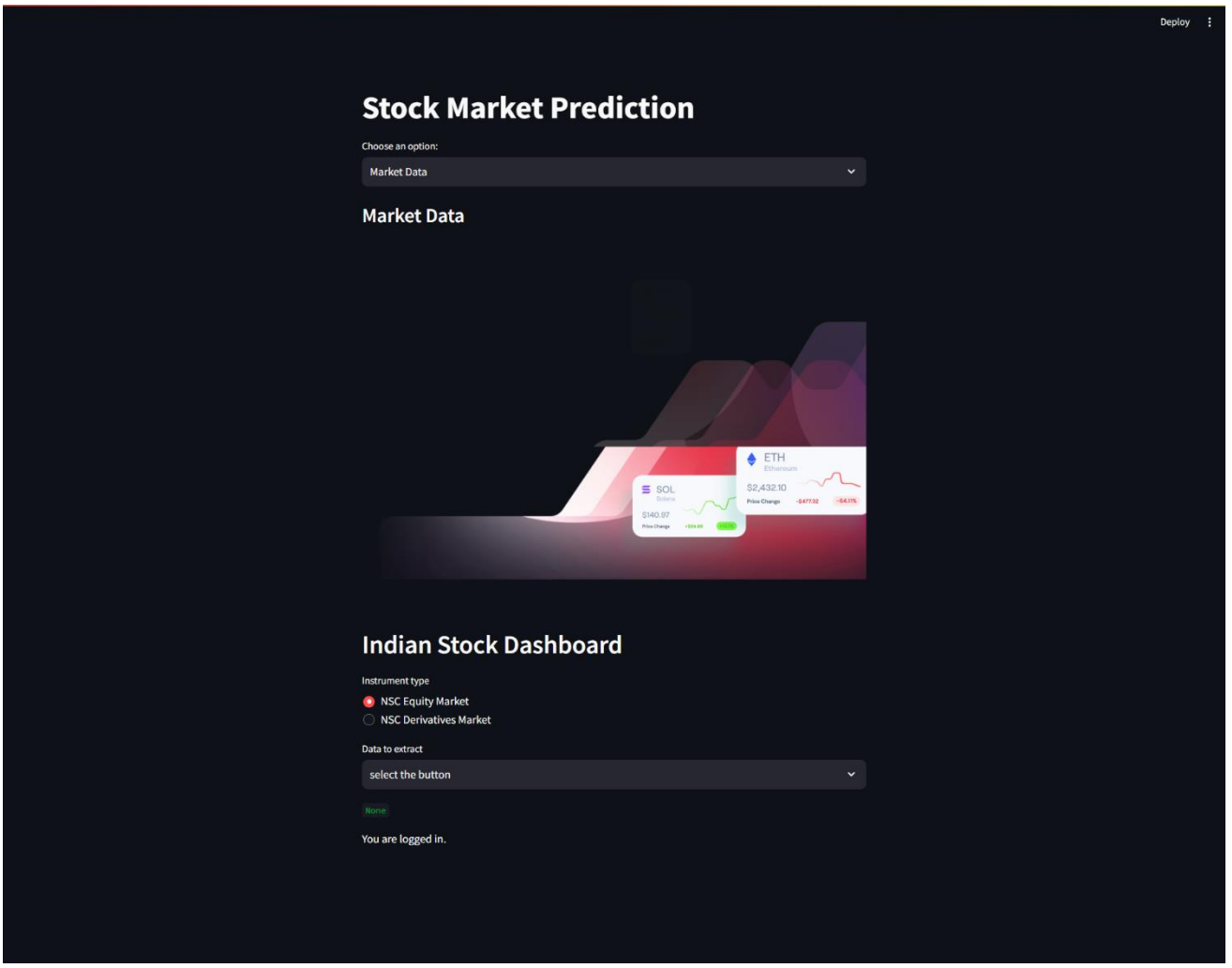


- This is the page of stock market prediction



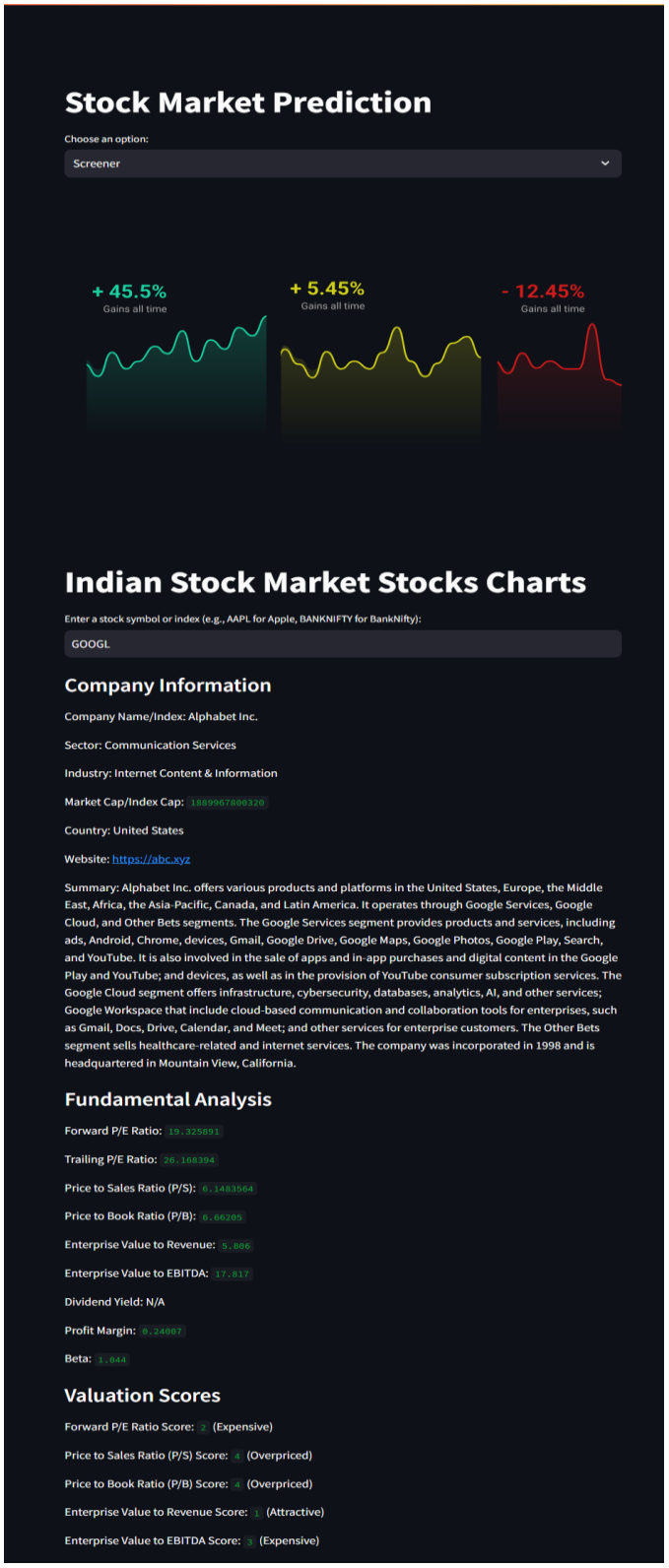


- This is the page for both login and signup
- The user login page with wrong credentials will show a message "You have entered wrong Username and Password



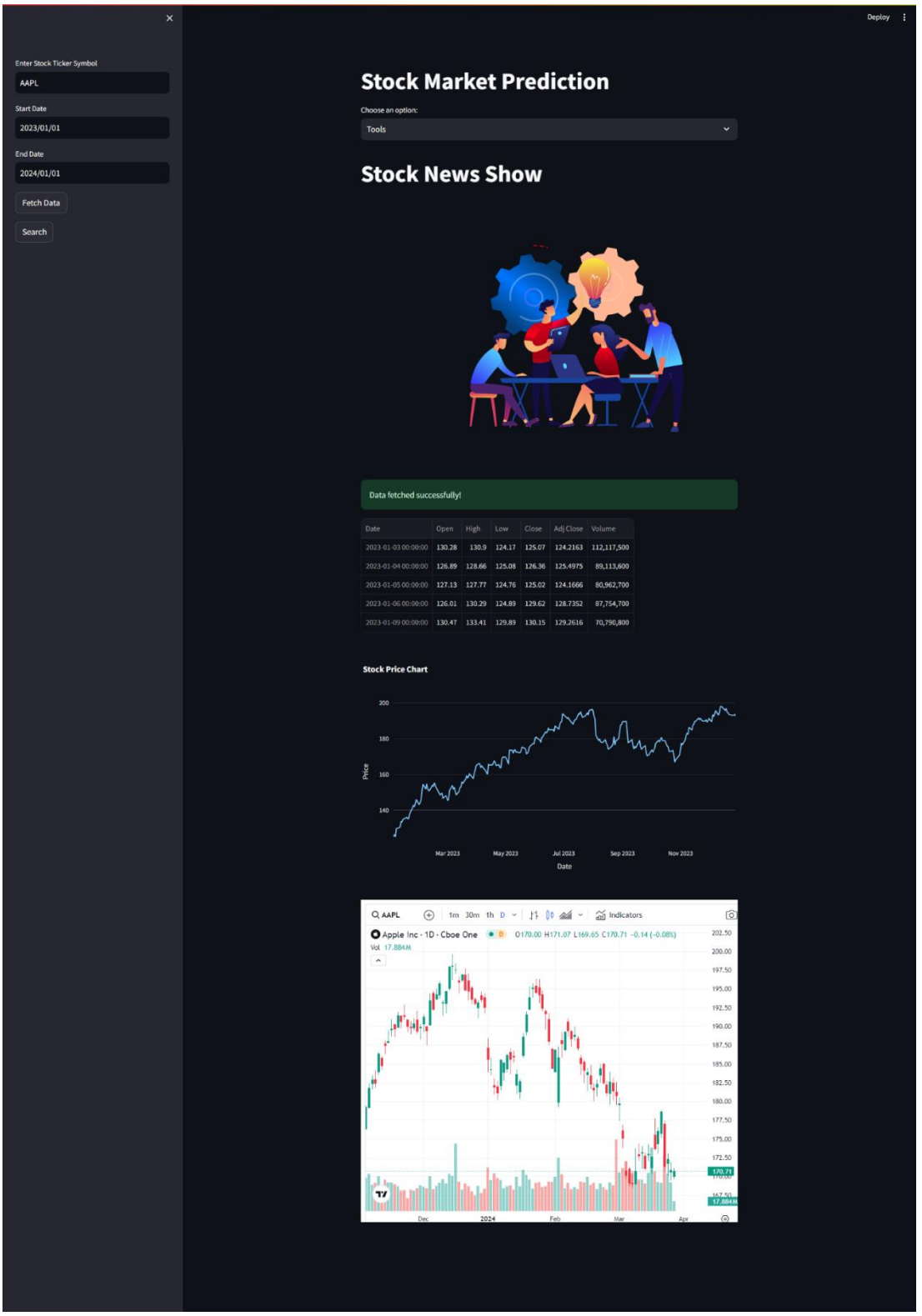
- This is our market data After Entering Correct Username and Password. This Homepage have Multiple Modules : Logout.

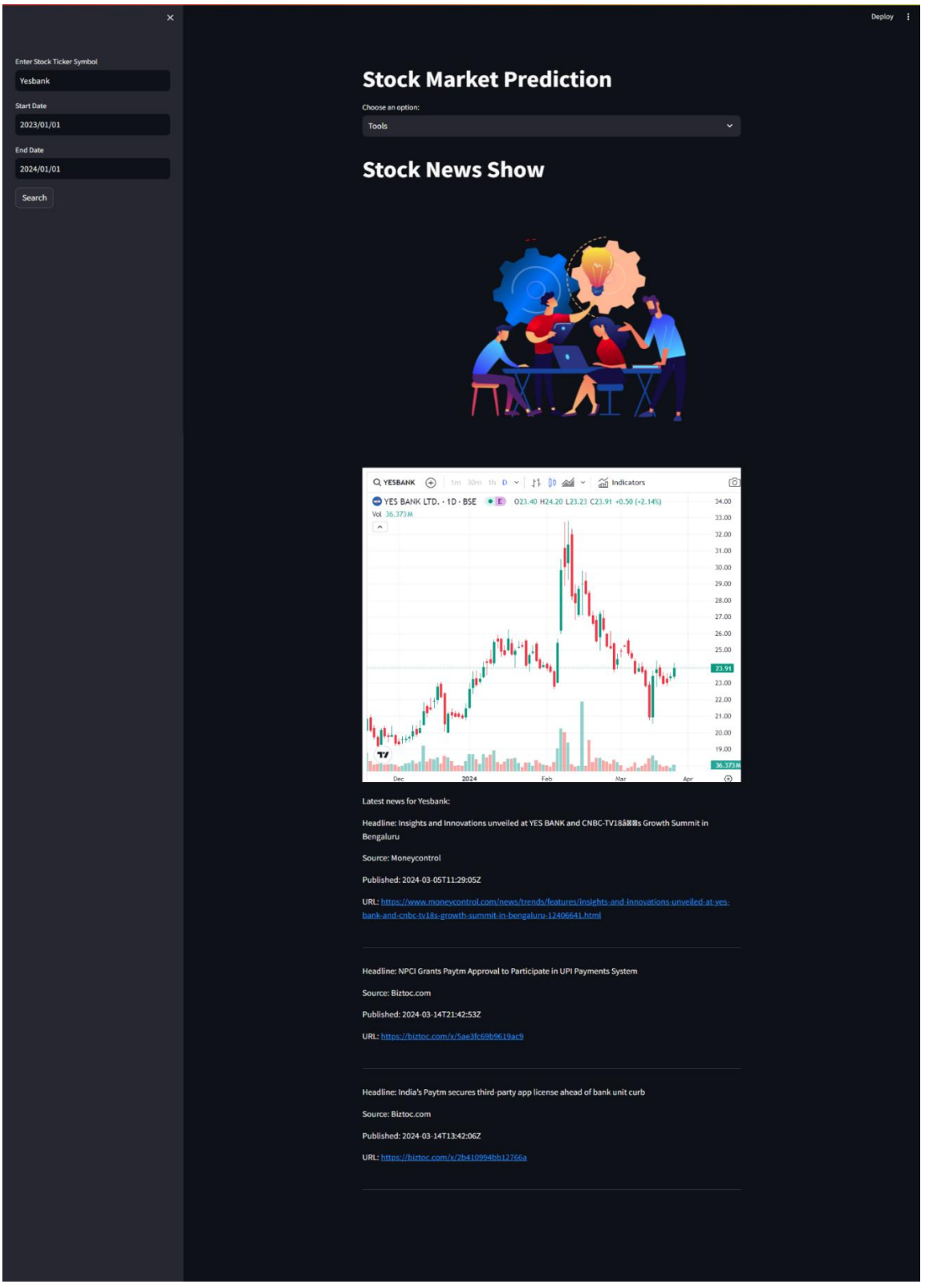
This is the screener page:-



- This page is all about company information , technical data and stock charts.

This is the tool page:-

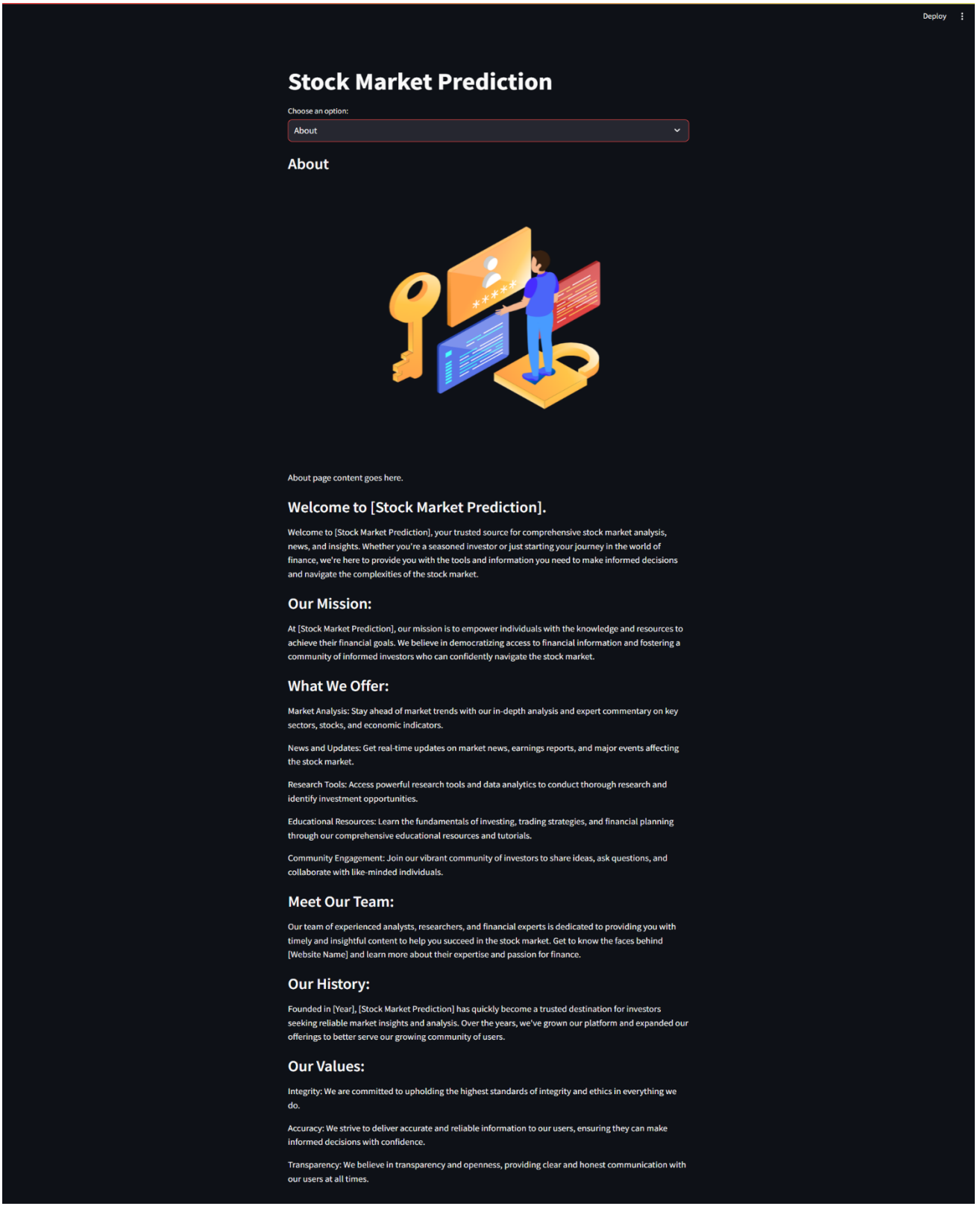




This Page Is All Aboubt Chart , Stock Data And News

This is Technical Analsis page

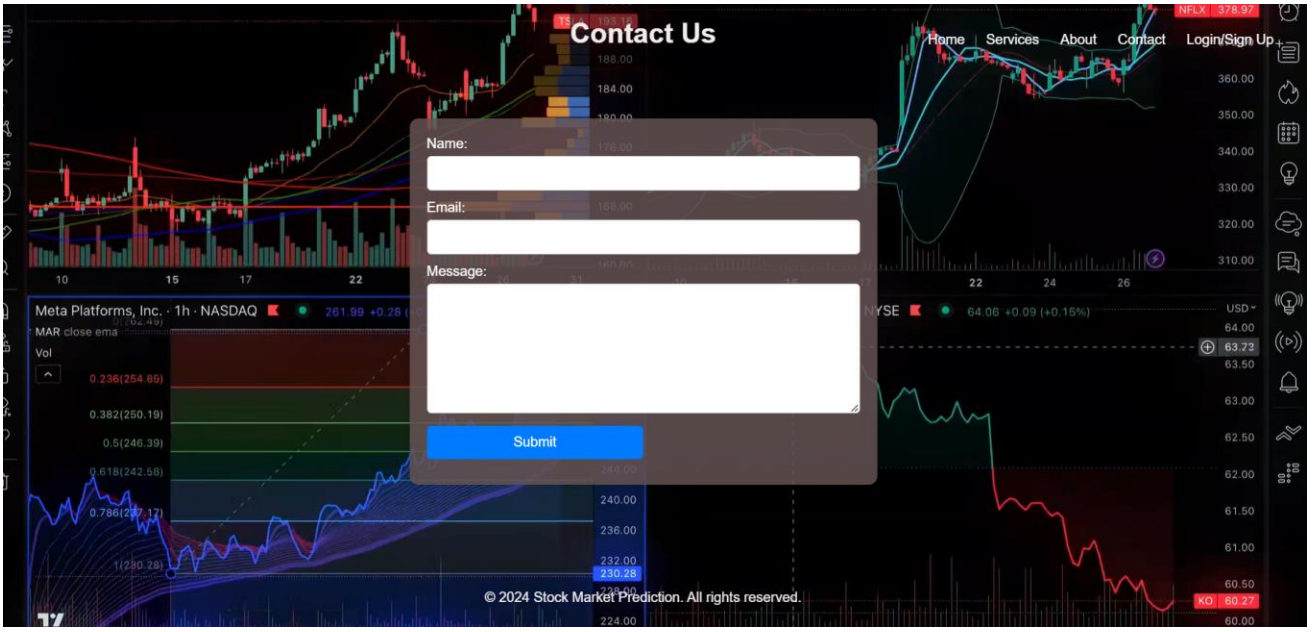
This Is Aboubt Page Of Stock Market Prediction



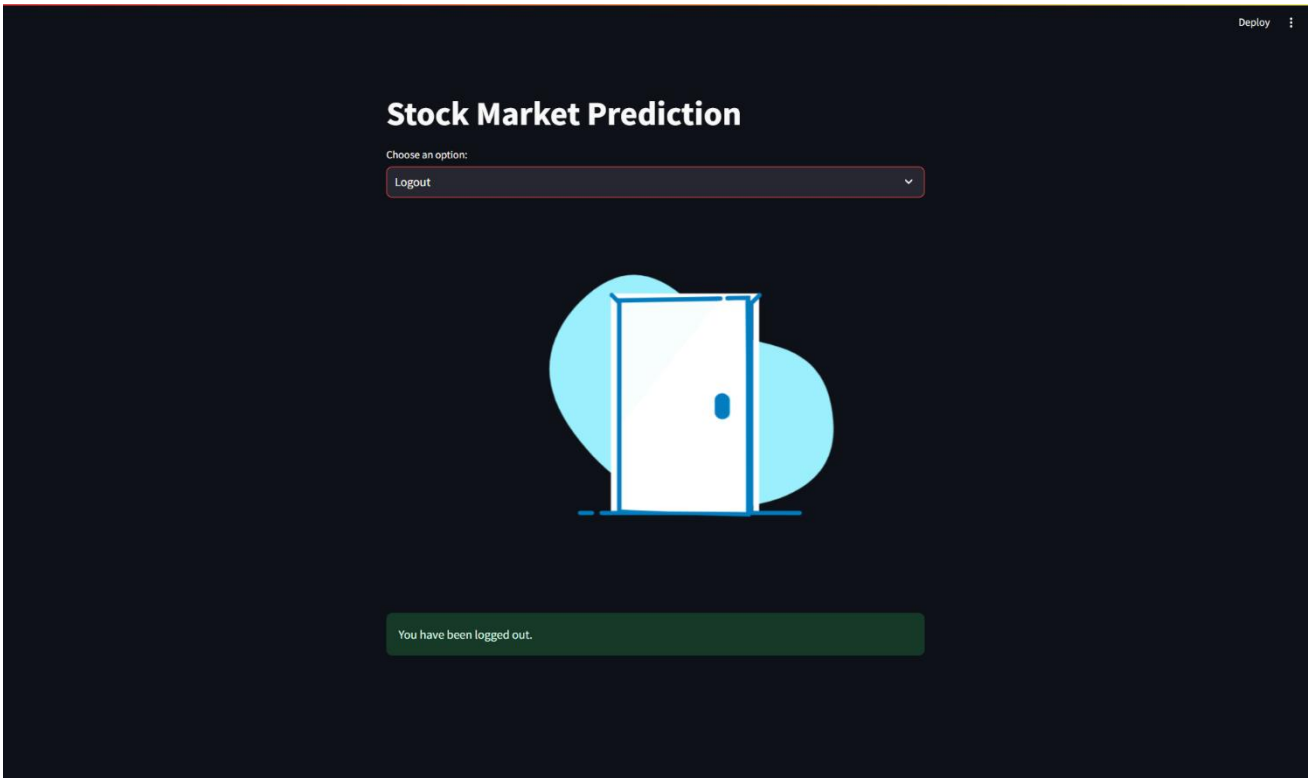
This Page Is Information About Website

This Is Contact Us Page

In Contact Us Page There Is Name, Email Message And Submit Button



After Clicking Submit Button You Will Get Log Out .



CONCLUSIONS

Conclusion

In the world of finance and investment, information is power. Making informed decisions in the stock market is crucial for investors, traders, and financial professionals. Over the past decades, stock market analysis and prediction systems have evolved significantly, offering tools and insights that empower stakeholders to navigate the complexities of the financial markets. This conclusion delves into the key takeaways and implications of such systems.

Stock market analysis and prediction systems have revolutionized how investors approach the financial markets. By harnessing vast amounts of historical and real-time data, these systems provide users with data-driven insights, enabling them to make informed investment decisions. Whether it's analyzing historical stock prices, tracking market trends, or predicting future price movements, these systems have become indispensable for investors seeking to maximize their returns and manage risk.

A cornerstone of stock market analysis and prediction systems is the application of advanced data analytics and machine learning techniques. These systems leverage historical market data to identify patterns, correlations, and trends that may be difficult for human analysts to discern. Machine learning models, such as neural networks and decision trees, can process vast datasets and generate predictions with remarkable accuracy. As a result, investors and financial professionals can gain a competitive edge by leveraging these cutting-edge technologies.

Challenges and Limitations:

While stock market analysis and prediction systems offer numerous benefits, they are not without challenges and limitations. One of the primary challenges is the inherent unpredictability of financial markets. Despite the sophistication of predictive models, factors like unexpected economic events, geopolitical developments, and market sentiment can lead to unpredictable price fluctuations. Additionally, data quality and model accuracy are critical factors that require ongoing attention and refinement.

Significance of the System

- User and Admin have different login activities for security.
- User can stock by area wise and category wise.
- The user can know the stock's information through the web application.
- Admin can configure the database and modify the data in it.

Limitations of the System

System Defects

the limitations of a "system defect" or potential issues with a system. However, your question is somewhat vague and lacks specific context some common limitations and issues that can be associated with various types of systems

- Little Memory for Storage
- Data Connection
- Battery Issues.
- Performance Problems.
- Software Issues.

Future Scope of the Project

There are also few features which can be integrated with this system to make it more flexible. Below list shows the future points to be consider

- Advancements in artificial intelligence and machine learning are poised to revolutionize stock market analysis by enabling more accurate predictions and real-time data analysis.
- The increasing availability of big data and historical market data will provide analysts with a wealth of information to develop more sophisticated predictive models.
- Algorithmic trading, powered by AI, will continue to gain popularity, leading to faster and more efficient execution of trades
- Robo-advisors and AI-driven investment platforms will become increasingly prevalent, making personalized investment advice more accessible to a broader range of investors.
- Portfolio Optimization: Developing strategies to diversify investments and manage risk, optimizing portfolios for maximum returns.
- Automated Trading: Implementing intelligent trading systems that can execute transactions based on real-time data analysis and decision-making algorithms.

REFERENCES

Books:-

- "Market Wizards" series by Jack D. Schwager
- "The Intelligent Investor" by Benjamin Graham
- "A Random Walk Down Wall Street" by Burton Malkiel
- "Stocks for the Long Run" by Jeremy Siegel

Website:-

- https://www.youtube.com/watch?v=fdFfpEtv5BU&ab_channel=FinancialProgrammingwithRitvik%2CCFA
- https://www.youtube.com/watch?v=YtxbFNKO5XU&t=891s&ab_channel=FinancialProgrammingwithRitvik%2CCFA
- https://www.youtube.com/watch?v=1fRRfVWxCVU&ab_channel=FinancialProgrammingwithRitvik%2CCFA
- https://www.youtube.com/watch?v=WiAGeVS6e40&t=285s&ab_channel=FinancialProgrammingwithRitvik%2 CCFA
- https://www.screener.in/
- https://streamlit.io/
- https://pypi.org/project/yfinance/
- https://colab.research.google.com/drive/1YcTArYNB6bN9HWQhmPsRfyqFftefG5yp#scrollTo=91vru17k7IFX
- https://lottiefiles.com/web-player
- https://www.nseindia.com/

Glossary:-

- **API-** Application Programme Interface
- **UML-** Unified Modelling Language
- **URL-** Uniform Resource Locator
- **SDK-** Software Development Kit
- **IPO -** Initial Public Offering
- **NSE** - National Stock Exchange of India