🚀 Cracking the Market Code with AI-Driven Stock Price Prediction Using Time Series Analysis

Phase-3 Submission

Student Name: HEMNATH S

Register Number: 412723104040

Institution: TAGORE ENGINEERING COLLEGE

Department: COMPUTERSCIENCE AND ENGINEERING

Date of Submission: 12/05/2025

Github Repository Link: https://github.com/Hemnath-07/stock-price-prediction-ai.git

---

1. Problem Statement

"The stock market is driven by a mix of economic trends and human emotion—can AI uncover the hidden patterns?"

The aim is to build a robust **AI-powered prediction system** that can accurately forecast **stock prices** using **Time Series Analysis** with models like **LSTM**, **GRU**, **Prophet**, and **ARIMA**. This empowers investors with data-backed decisions beyond traditional charts and gut feelings.

2. Abstract

This project tackles the volatility of stock markets using advanced AI-driven methods. It aims to accurately forecast future stock prices and classify market trends through time series analysis. Using deep learning models such as LSTM and GRU, along with classical models like Prophet and ARIMA, the system processes both static and live stock data. The pipeline covers data collection, preprocessing, modeling, and deployment through a Streamlit web app. The model's predictions are visualized via interactive dashboards, offering insightful analytics for traders and investors. The solution is open-source and deployable, allowing real-time financial forecasting with high interpretability.

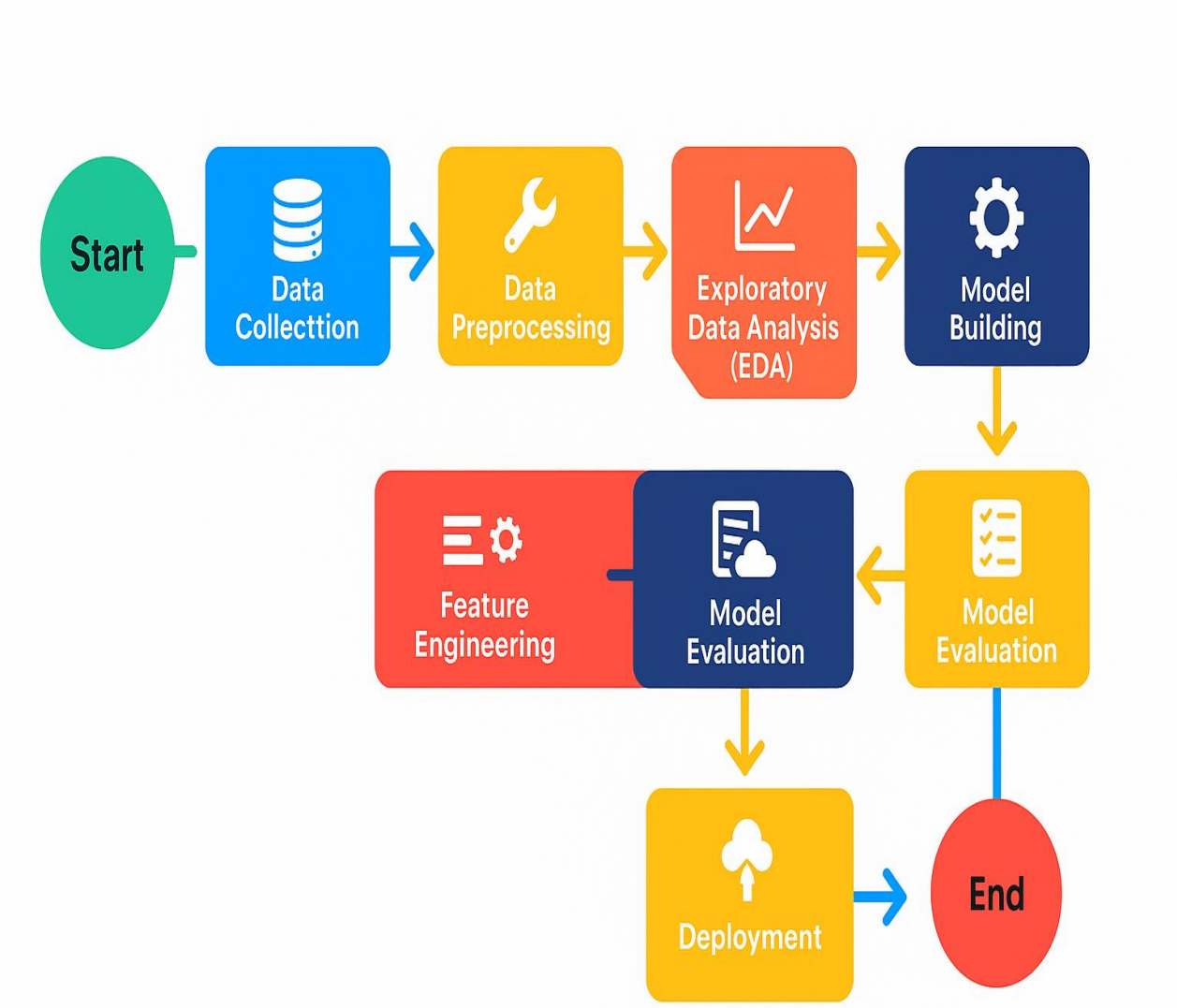
3. System Requirements

- **Hardware:** 8GB+ RAM (minimum), i3 or equivalent for local runs
- **Software:** Python 3.10+, Google Colab, Streamlit Cloud
- **Libraries:** *tensorflow, prophet, yfinance, scikit-learn, matplotlib, seaborn, plotly,* etc.

4. Objectives

- Predict short- and long-term stock prices
- Classify trends (bullish/bearish)
- Compare ML/DL models
- Deploy results using an interactive UI

5. Flowchart of Project Workflow



6. Dataset Description

- ✓ Kaggle (Nifty50, S&P 500)
- ✓ Yahoo Finance (via yfinance)
  Structure: Date, Open, High, Low, Close, Volume (Target: Close)

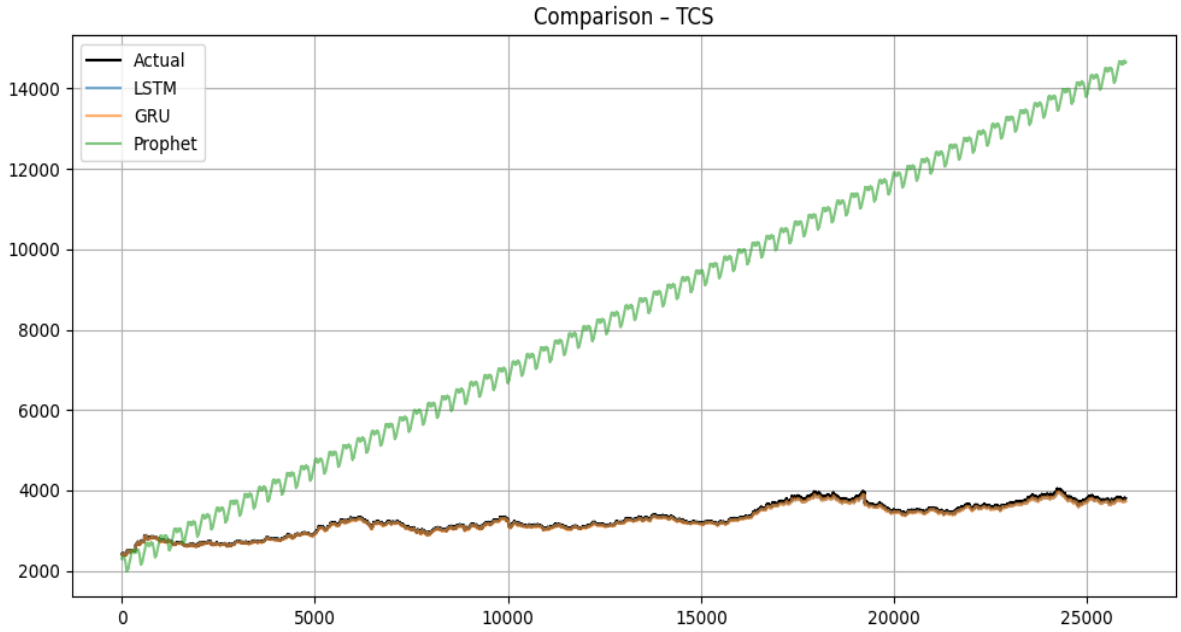
7. Data Preprocessing

- Null removal, type casting
- 60-day sequence framing for LSTM
- Scaling using `MinMaxScaler`
  Include before/after screenshots

1	Date	unnamed: 1	unnamed: 2	unnamed: 3	unnamed: 4	unnamed: 5
2	2019-01-02	38.72249984741211	39.712501525878906	38.557498931884766	39.47999954223633	148158800
3	2019-01-03	35.994998931884766	36.43000030517578	35.5	35.54750061035156	365248800
4	2019-01-04	36.13249969482422	37.13750076293945	35.95000076293945	37.06499862670898	234428400
5	2019-01-07	37.17499923706055	37.20750045776367	36.47499847412109	36.98249816894531	219111200
6	2019-01-08	37.38999938964844	37.95500183105469	37.130001068115234	37.6875	164101200
7	2019-01-09	37.8224983215332	38.63249969482422	37.407501220703125	38.32749938964844	180396400
8	2019-01-10	38.125	38.49250030517578	37.71500015258789	38.45000076293945	143122800
9	2019-01-11	38.220001220703125	38.42499923706055	37.87749862670898	38.0724983215332	108092800
10	2019-01-14	37.712501525878906	37.817501068115234	37.30500030517578	37.5	129756800

8. Exploratory Data Analysis (EDA)

- ❖ Line plots of price
- ❖ Volume charts
- ❖ Moving averages
- ❖ Heatmaps



9. Feature Engineering

- ➢ Technical indicators (SMA, EMA, RSI)
- ➢ Trend labels
- ➢ Sliding windows for time steps
- ➢ Reasoning: These enhance temporal understanding for DL mode

10. Model Building

- Compare models:
- LSTM & GRU (deep learning)
- Prophet (decomposition model)

```
🚀 Preprocessing: ITC

🚀 Preprocessing: AAPL

🗓️ All stocks preprocessed and saved. Models will be trained LIVE.
```

11. Model Evaluation

- *Metrics: RMSE, MAE, MAPE*
- *Include confusion matrix (if doing trend classification)*
- *Visualize actual vs predicted values*

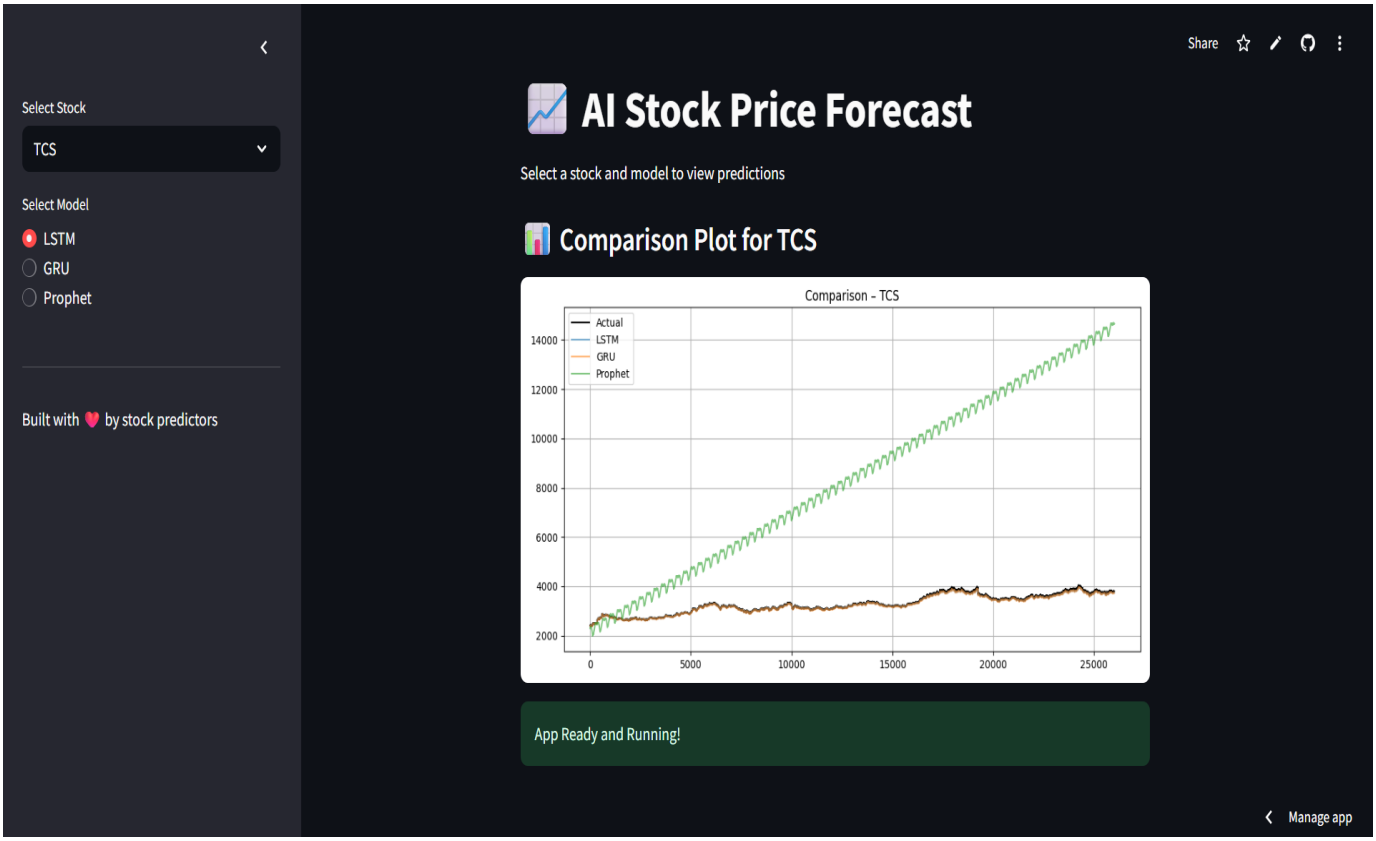
```
🚀 Training Models for: ITC
813/813 ━━━━━━━━━━━━━━━━━━━━ 7s 8ms/step
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.kera
813/813 ━━━━━━━━━━━━━━━━━━━━ 8s 9ms/step
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.kera
DEBUG:cmdstanpy:input tempfile: /tmp/tmpbp9blb_d/rebewhgg.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpbp9blb_d/_x6kkore.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.11/dist-packages/prophet/stan_model/prophet_model.bin', 'random', 'seed=18474', 'data', 'file=/tmp/tmpbp9blb_d/rebewhgg.json', 'init=/tmp/tmpbp9blb_d/_x6kkore.json', 'output', 'f
12:28:57 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
12:34:26 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
  ✅ Done: LSTM RMSE=0.52, GRU RMSE=0.40, Prophet RMSE=2465.4261586426346

🚀 Training Models for: AAPL
8/8 ━━━━━━━━━━━━━━━━━━━━ 1s 44ms/step
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.kera
WARNING:tensorflow:5 out of the last 822 calls to <function TensorFlowTrainer.make_predict_function.<locals>.one_step_on_data_distributed at 0x7d5a30654860> triggered tf.function retracing. Tracing is expensive and the excessive numb
8/8 ━━━━━━━━━━━━━━━━━━━━ 1s 50ms/step
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.kera
  ⚠️ Prophet failed for AAPL: [Errno 2] No such file or directory: 'data/processed/enriched/static/AAPL_live.csv'
  ✅ Done: LSTM RMSE=9.32, GRU RMSE=6.10, Prophet RMSE=N/A

🎖️ All models trained and plots saved.
```

12. Deployment

Deployed on **Streamlit Cloud**

Public link : https://github.com/Hemnath-07/stock-price-prediction-ai.git

UI Screenshot :



Sample prediction output :

13. Source code

Public link to source code : **https://github.com/Hemnath-07/stock-price-prediction-ai.git**

14. Future scope

- **Integration of Sentiment Analysis**

Future iterations of the system can incorporate sentiment analysis using data from financial news articles, Twitter, or Reddit. This would allow the model to capture the impact of market sentiment and investor emotion on stock prices, improving prediction accuracy.

- **Adoption of Transformer-Based Models (e.g., BERT, TCN, or Informer)**

Emerging deep learning models like Transformers and Temporal Convolutional Networks (TCNs) have shown promise in time series tasks. These models can be explored to capture long-range dependencies more efficiently than LSTM/GRU.

- **Portfolio Optimization Module**

Extend the application to suggest optimal stock portfolios based on risk-return profiles and AI-driven forecasts, enabling smarter investment strategies for users.

- **Multi-Asset and Sector-Based Analysis**

The system can be expanded to analyze entire sectors (e.g., IT, Pharma) and other financial instruments such as cryptocurrencies or commodities to serve a wider range of investors.

- **AutoML & Hyperparameter Optimization Integration**

Incorporating AutoML platforms or Bayesian optimization can automate model tuning and selection, saving time and enhancing performance.

- **Mobile App Development**

Building a mobile version of the app can allow investors to access real-time predictions and trend classifications on the go.

13. Team Members and Roles

NAME	ROLE
ADITHYA B	End-to-end development, modeling
HEMNATH S	Model evaluation & tuning
HARISHKUMAR K	REPORT & PRESENTATION
JOSHUVA D	DATA COLLECTION