Github Link: https://github.com/Gokul9042/Gokul/blob/main/README.md

PHASE-2

“Cracking the Market Code with AI-driven Stock Price Prediction Using Time Series Analysis”:

1. Problem Statement

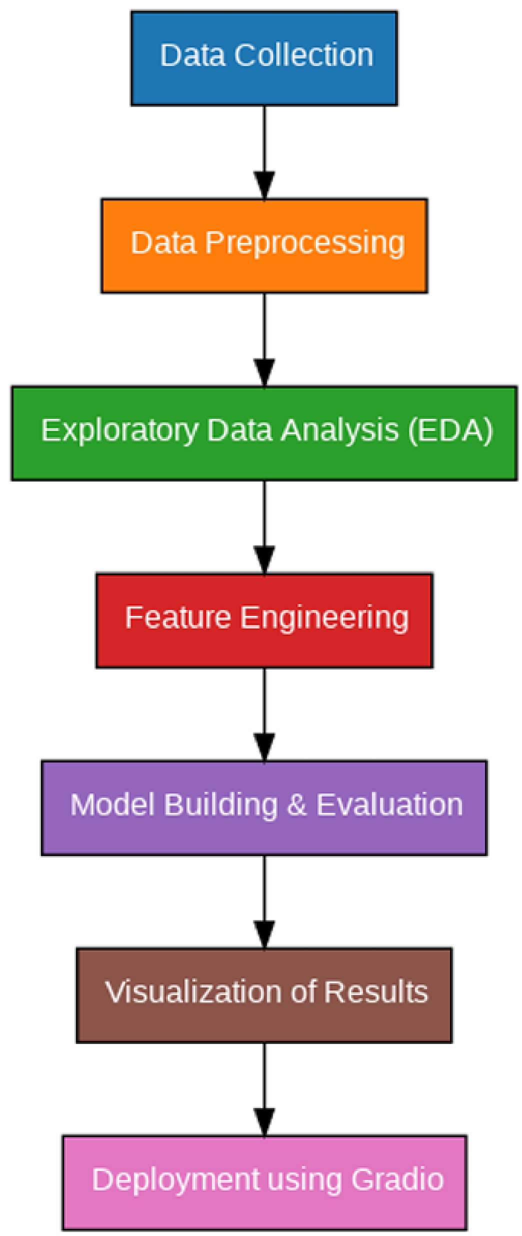
Predicting stock prices is a central challenge in the financial industry due to the volatile and non-linear nature of financial markets. The objective of this project is to develop an AI-driven model that can forecast future stock prices based on historical trends using time series analysis. Accurate forecasting can support investors, analysts, and portfolio managers in making data-driven decisions. This project uses machine learning and deep learning techniques to model sequential patterns in stock data.

The problem is framed as a **regression task**, where the goal is to predict future stock prices (e.g., next-day closing price) using a sequence of past prices and technical indicators.

2. Project Objectives

- Develop a machine learning/deep learning model that predicts future stock prices using time series data.
- Apply technical indicators (e.g., moving averages, RSI) to enhance predictive performance.
- Compare classical time series models (e.g., ARIMA) with AI models (LSTM, GRU).
- Ensure interpretability and evaluate model reliability for financial applications.
- Provide a user-friendly interface via Gradio for real-time predictions.
- **Evolved Goal:** After initial exploration, the focus was refined to use deep learning (LSTM) due to its superior performance on sequential data.

3. Flowchart of the Project Workflow



4. Data Description

- **Dataset Name:** Historical Stock Market Data (e.g., AAPL, S&P 500)
- **Source:** Yahoo Finance via `yfinance` API
- **Type of Data:** Time series (date-indexed tabular)
- **Records and Features:** ~2,000+ records, with features such as Open, High, Low, Close, Volume
- **Target Variable:** Next-day closing price (continuous)
- **Static or Dynamic:** Dynamic (data updates over time)
- **Attributes Covered:** OHLC prices, Volume, and derived technical indicators

5. Data Preprocessing

- Checked for and handled missing timestamps (e.g., weekends, holidays).
- Filled null values using forward-fill/backward-fill strategies.
- Applied log transformation and differencing to stabilize variance (for classical models).
- Normalized numeric features using MinMaxScaler for neural networks.
- Ensured stationarity where required (ADF test for ARIMA).

6. Exploratory Data Analysis (EDA)

- **Univariate Analysis:**
    - Time series plots for price trends
    - Histograms of price returns
- **Bivariate/Multivariate Analysis:**
    - Correlation matrix between indicators and price
    - Line plots comparing moving averages with actual price
- **Key Insights:**
    - Strong auto-correlation observed in short lags (ACF/PACF)
    - Price tends to revert after sharp increases (mean reversion patterns)
    - Momentum indicators (MACD, RSI) show predictive potential

7. Feature Engineering

- Created lag features (e.g., Close_1, Close_2,...Close_n)
- Derived rolling metrics: Moving Averages (MA5, MA10), Bollinger Bands
- Engineered momentum indicators: RSI, MACD
- Time-based features: day of week, month, trading volume trends
- Framed supervised learning structure using sliding window technique

8. Model Building

- **Algorithms Used:**
    - **ARIMA**: Baseline classical method for time series forecasting
    - **LSTM (Long Short-Term Memory):** Deep learning model tailored for sequential dependencies
- **Model Selection Rationale:**
    - ARIMA: Good for linear temporal trends
    - LSTM: Captures long-term dependencies and non-linear patterns
- **Train-Test Split:**
    - Used **80/20** chronological split to avoid data leakage
    - Data reshaped into 3D arrays for LSTM input
- **Evaluation Metrics:**

- **MAE (Mean Absolute Error)**
- **RMSE (Root Mean Squared Error)**
- **R^2 Score** for overall trend accuracy

9. Visualization of Results & Model Insights

- **Prediction Plots:**
  - Actual vs Predicted closing prices on test set
- **Model Comparison:**
  - ARIMA vs LSTM (LSTM showed significantly lower RMSE and better trend capture)
- **Residual Analysis:**
  - Plots confirmed that LSTM had less bias and variance in error terms
- **Feature Impact:**
  - LSTM attention focused on recent price lags and RSI as key indicators
- **User Testing:**
  - Built Gradio UI for real-time input of last 10 closing prices + indicators
  - Displays predicted price with confidence intervals

10. Tools and Technologies Used

- **Programming Language:** Python 3
- **Notebook Environment:** Google Colab & Jupyter
- **Key Libraries:**
  - `pandas`, `numpy`: Data processing
  - `matplotlib`, `seaborn`, `plotly`: Visualization
  - `scikit-learn`: Preprocessing, metrics
  - `statsmodels`: ARIMA modeling
  - `tensorflow.keras`: LSTM modeling
  - `Gradio`: Deployment of web-based interface
  - `yfinance`: Data acquisition from Yahoo Finance

11. Team Members and Contributions

- **R.Gokul –** Data Collection and Preprocessing
- **R.Hariprasanth–** Feature Engineering and Time Series Framing
- **JageshRajoo–** LSTM Model Development and Evaluation