Investment Prediction System

Written By	Suraj Kakulte
Document Version	1.0
Last Revised Date	06/06/2025

Document Control

Change Record:

Version	Date	Author	Comments
1.0	06/06/2025	Suraj Kakulte	Initial Low-Level Design document

Reviews:

Version	Date	Reviewer	Comments

Approval Status:

Version	Review Date	Reviewed By	Approved By	Comments

Contents

1. Introduction

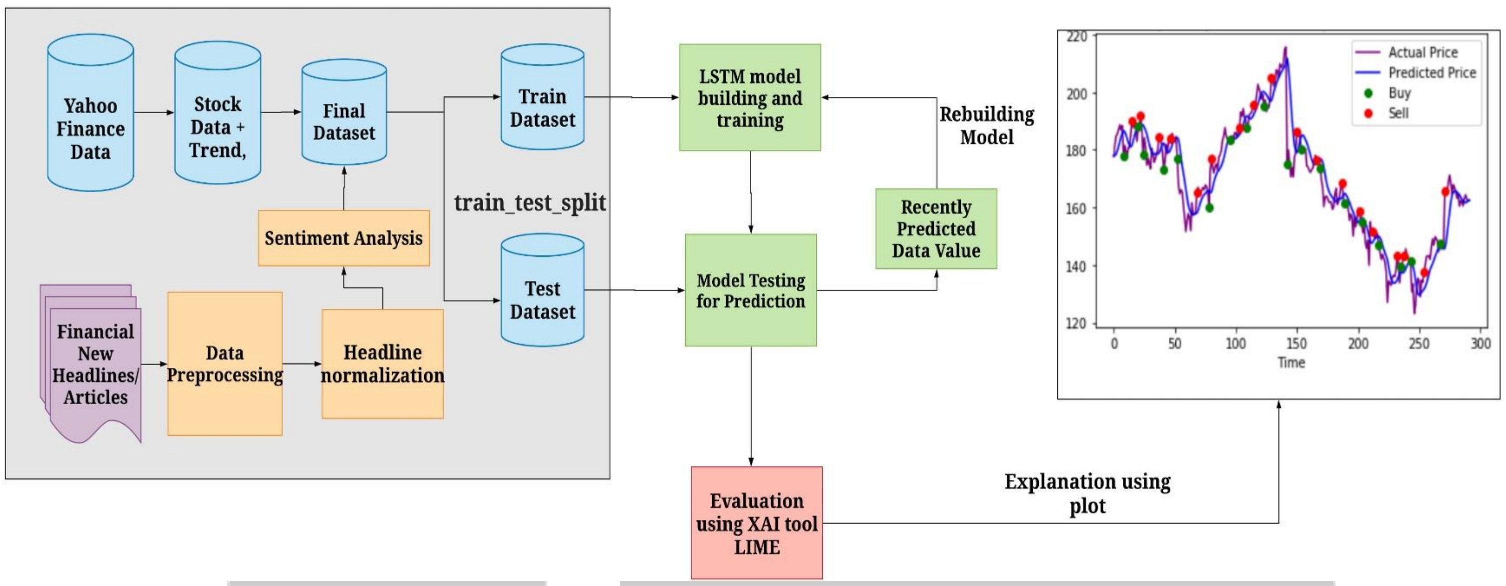
1.1. What is Low-Level design document?

The Low-Level Design (LLD) document outlines the internal components and detailed design of the Stock Market Prediction Web App. It explains data flow, module relationships, class structures, and function descriptions to aid in coding and integration.

1.2. Scope

This LLD document focuses on the implementation-level design for data collection, preprocessing, model training, sentiment analysis, user interface integration, and backend prediction logic. It ensures the software components are precisely defined and ready for development.

2. Architecture



3. Architecture Description

3.1. Data Description

- Stock Data Source: Historical data from Yahoo Finance and Alpha Vantage APIs.
- Tweet Data Source: Twitter API (Tweepy) used for sentiment analysis.
- Data Format: JSON, CSV

3.2. Data Collection

- Collect stock prices (Open, Close, High, Low, Volume).
- Scrape or retrieve latest tweets by ticker symbol.
- Collect financial news (optional module).

3.3. Data Transformation

- Convert API results into structured Pandas Data Frames.
- Standardize column names, format datetime objects, and remove duplicates.

3.4. Data Insertion into Database

- **Database Used:** MySQL
- Create tables for:
  - Users
  - Predictions
  - Tweets
  - Logs
  - Convert API results into structured Pandas Data Frames.

3.5. Export Data from Database

- Export historical user prediction records and ML logs for retraining or debugging.

3.6. Data Pre-processing

- **For Stock Data:**
  - Handle missing values
  - Normalize data
- **For Tweets:**
  - Remove stopwords, hashtags, punctuation
  - Tokenize and apply sentiment scoring (TextBlob or VADER)

3.7. Sentiment Analysis

- Score tweets as positive, neutral, or negative.
- Aggregate score over time to create sentiment trend vectors.

3.8. Prediction Interference

- Load models dynamically when a prediction is requested.
- Adjust model prediction using recent sentiment trends.
- Output forecasted prices for 7 future days.

3.9. Model Training

- Train LSTM and ARIMA models on historical data.
- Store trained models (.h5 or .pkl files) for inference.

3.10. Data from User

- Accepts stock ticker input from authenticated users via frontend.

3.11. Data Validation

- Validate user input (correct symbol, non-empty).
- Check API responses and data integrity.

3.12. User Data Inserting into Database

- Store prediction history, user ID, timestamp, and forecast results.

3.13. Result Display on Dashboard

- Display predicted vs. actual prices in line charts.
- Show sentiment polarity over recent days.
- Present related news or tweets (optional feature).

3.14. Deployment

- Use XAMPP for local WordPress + Flask integration.
- Python scripts hosted via Flask app (main.py).
- WordPress frontend connects via REST endpoints or iframe embedding.

4. Unit Test Cases

Test Case Description	Pre-Requisite	Expected Result
Verify application loads for user	App URL is defined	Page loads successfully
Verify user registration and login	App is accessible	User can register/login
Validate stock ticker input	User is logged in	Accepts valid tickers, rejects invalid ones
Verify prediction output appears	Valid input provided	Returns 7-day forecast and sentiment chart
Verify tweets are fetched	Twitter API keys are valid	Recent tweets are displayed and scored
Validate error handling for API failure	Disconnect API intentionally	Shows fallback/error message
Check if data is inserted into prediction history table	Prediction made	Data is logged correctly in MySQL
Check if sentiment affects prediction	Vary tweet polarity	Forecast adjusts with sentiment weight
Verify visualization of price & sentiment	Predictions completed	Charts are visible and interactive