Investment Prediction System

Architecture Documentation

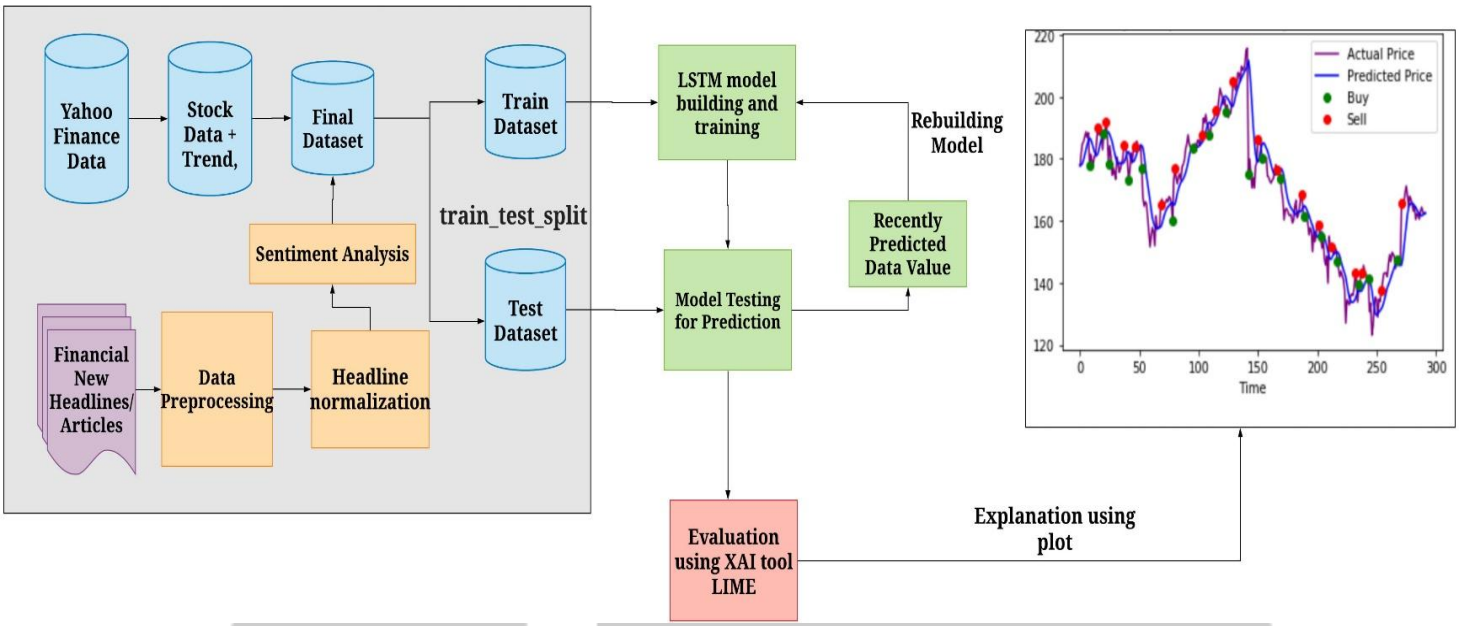
Revision Number: 1.0

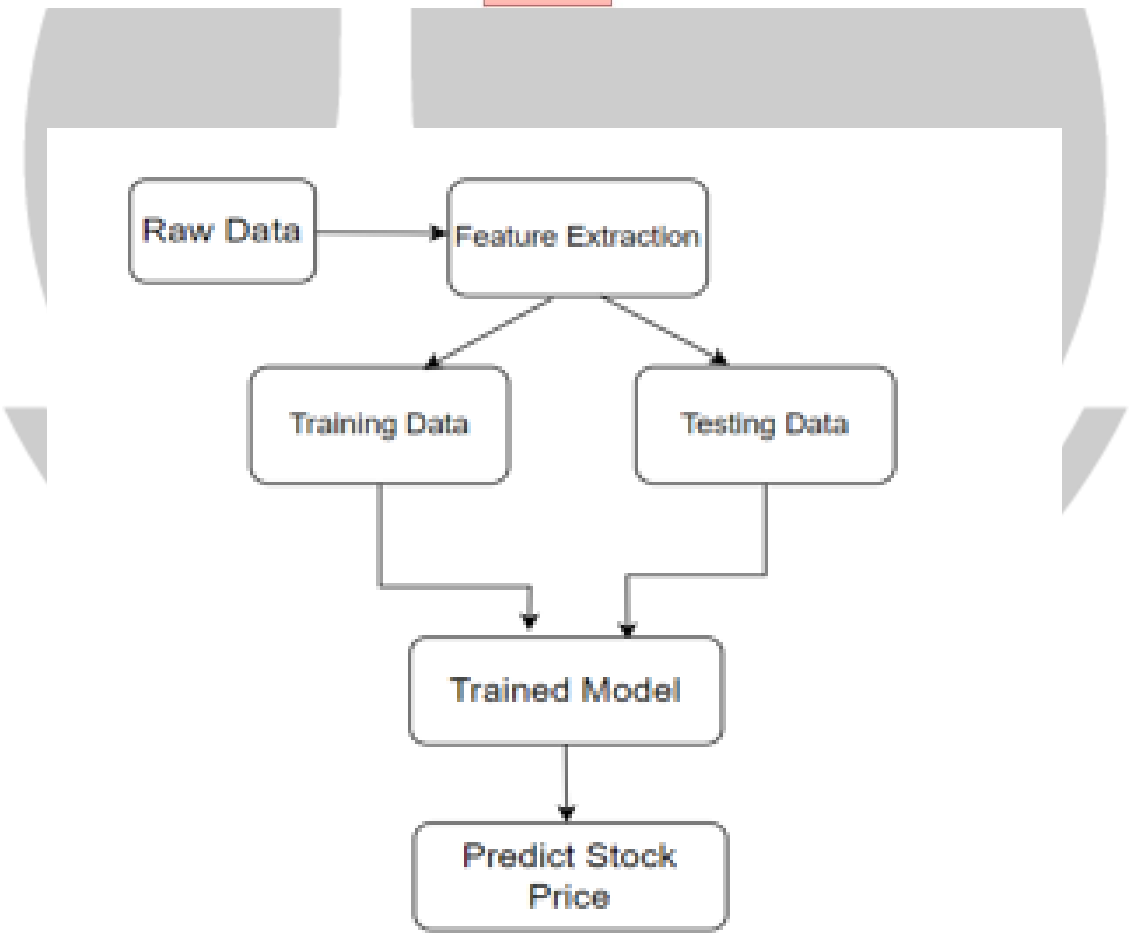
Last date of revision: 06/06/2025

Investment Prediction System

Date Issued	Version	Description	Author
06/06/2025	1	Initial Architecture Document	Suraj Kakulte

Architecture





1. Architecture Description

1.1. 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

1.2. Data Collection

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

1.3. Data Transformation

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

1.4. Data Insertion into Database

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

1.5. Export Data from Database

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

1.6. 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)

1.7. Sentiment Analysis

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

1.8. Prediction Interference

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

1.9. Model Training

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

1.10. Data from User

- Accepts stock ticker input from authenticated users via frontend.

1.11. Data Validation

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

1.12. User Data Inserting into Database

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

1.13. 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).

1.14. 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.