



EXPENSEXCHANGE AI

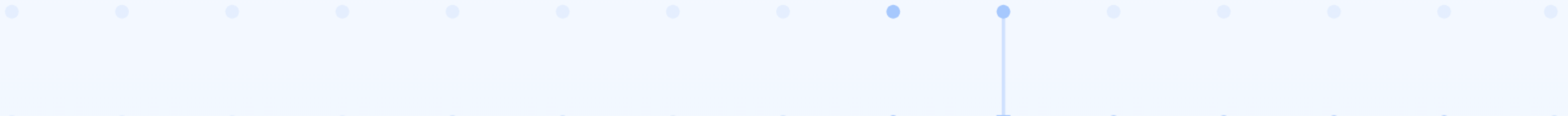
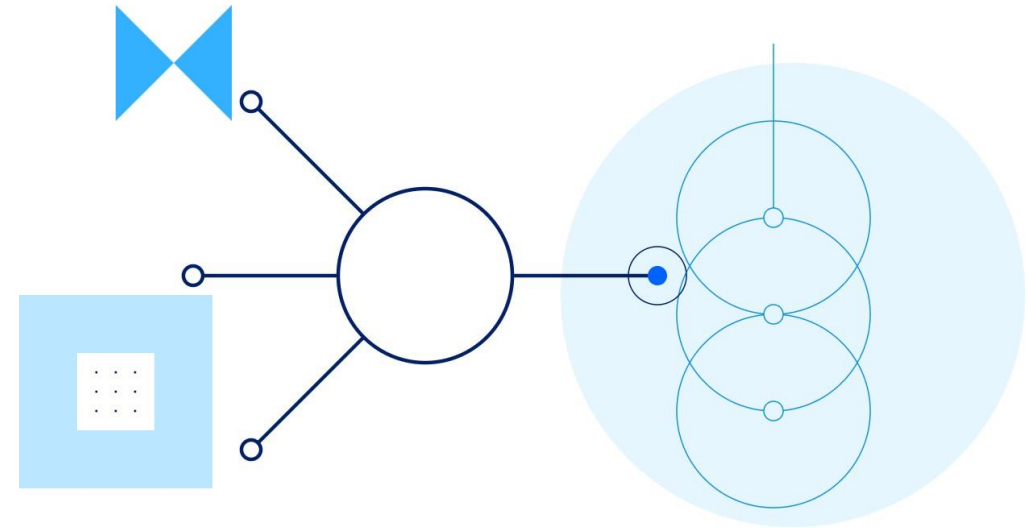
Team Members:

1. Prince Yadav, 500108278, 6, AI&ML
2. Sparsh, 500101785, 8, AI&ML
3. Dakshita Jindal, 500108380, 1, AI&ML
4. Anshu Singh, 500102134, 7, AI&ML

Content



1. Abstract
2. Introduction
3. Problem Statement
4. Objective
5. Methodology
6. LSTM model
7. GRU model
8. Applications





Abstract

This project presents a stock return analysis system that uses a Gated Recurrent Unit (GRU) model to forecast short-term returns based on historical price data. The system includes a deep learning model, a Flask-based backend, and a browser-accessible frontend. Users can upload their current holdings and receive return predictions along with basic investment suggestions. The model operates on log-returns and is trained using five years of historical data. Optuna tunes the model's hyperparameters, and Huber loss guides training to reduce the impact of outliers. The system evaluates performance using standard regression and finance-specific metrics, including mean absolute error, root mean square error, R^2 score, and directional accuracy. While the system does not offer exact price predictions or guaranteed outcomes, it provides a working example of how machine learning can assist in financial decision-making.



Introduction



- This project explores the use of deep learning, specifically Gated Recurrent Units (GRUs), to model short-term stock price movements based solely on historical closing prices. By excluding external data such as news or technical indicators, the system remains focused, reproducible, and easier to interpret. To improve predictive performance, key model parameters like sequence length, learning rate, and dropout are optimized using Optuna. Huber loss is employed during training to ensure robustness against market outliers.
- The trained GRU models are deployed through a Flask backend and accessed via a user-friendly frontend. Users can upload their stock portfolios to receive short-term return forecasts, which are translated into simple action suggestions—buy, hold, or sell—based on predicted price trends. While the system is not intended to offer financial advice, it demonstrates a complete and functional machine learning pipeline, from raw data to practical output, in the context of stock analysis.





Problem Statement

Predicting short-term stock price movements is inherently challenging due to market volatility, limited data visibility, and the influence of unpredictable external events. Traditional models often rely on complex feature engineering or incorporate external indicators that reduce reproducibility. This project aims to address the problem of forecasting short-term log-returns of individual stocks using only historical price data. The goal is to build a reliable, lightweight deep learning system—based on Gated Recurrent Units (GRUs)—that can identify patterns in past trends and provide actionable insights to support basic portfolio decisions. The solution must be efficient, interpretable, and deployable as a complete end-to-end machine learning pipeline.





Objectives

- Forecast short-term stock returns using a GRU-based model trained on historical price data and log-returns.
- Translate predictions into simple portfolio actions—buy, hold, or sell—based on expected return direction.
- Provide a user-facing interface where users can upload their portfolio and receive model-based feedback.
- Demonstrate a complete and modular machine learning system that moves from raw financial data to interpretable output



Methodology



Data Preprocessing:

Loaded and cleaned historical stock data (say 7 years).
Kept only 'Date' and 'Close' columns; sorted by date.
Normalized prices using MinMaxScaler.

Train-Test Split:

Split dataset into training (70%) and testing (30%) sets.

Model Selection:

Used LSTM model and GRU model with sequential architecture.

Model Training:

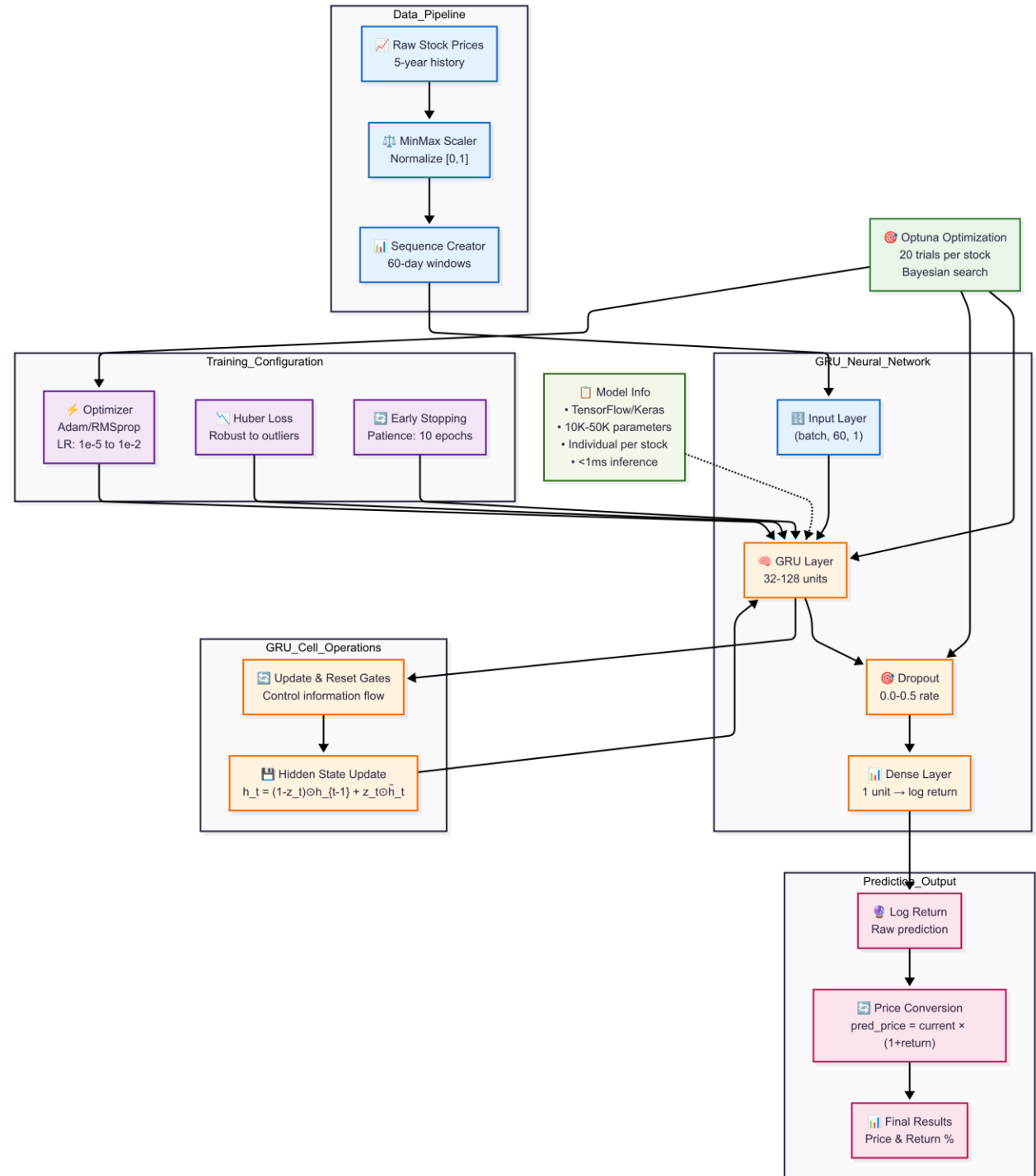
Trained separately for TCS, Infosys, and L&T using MSE loss.

Prediction Output:

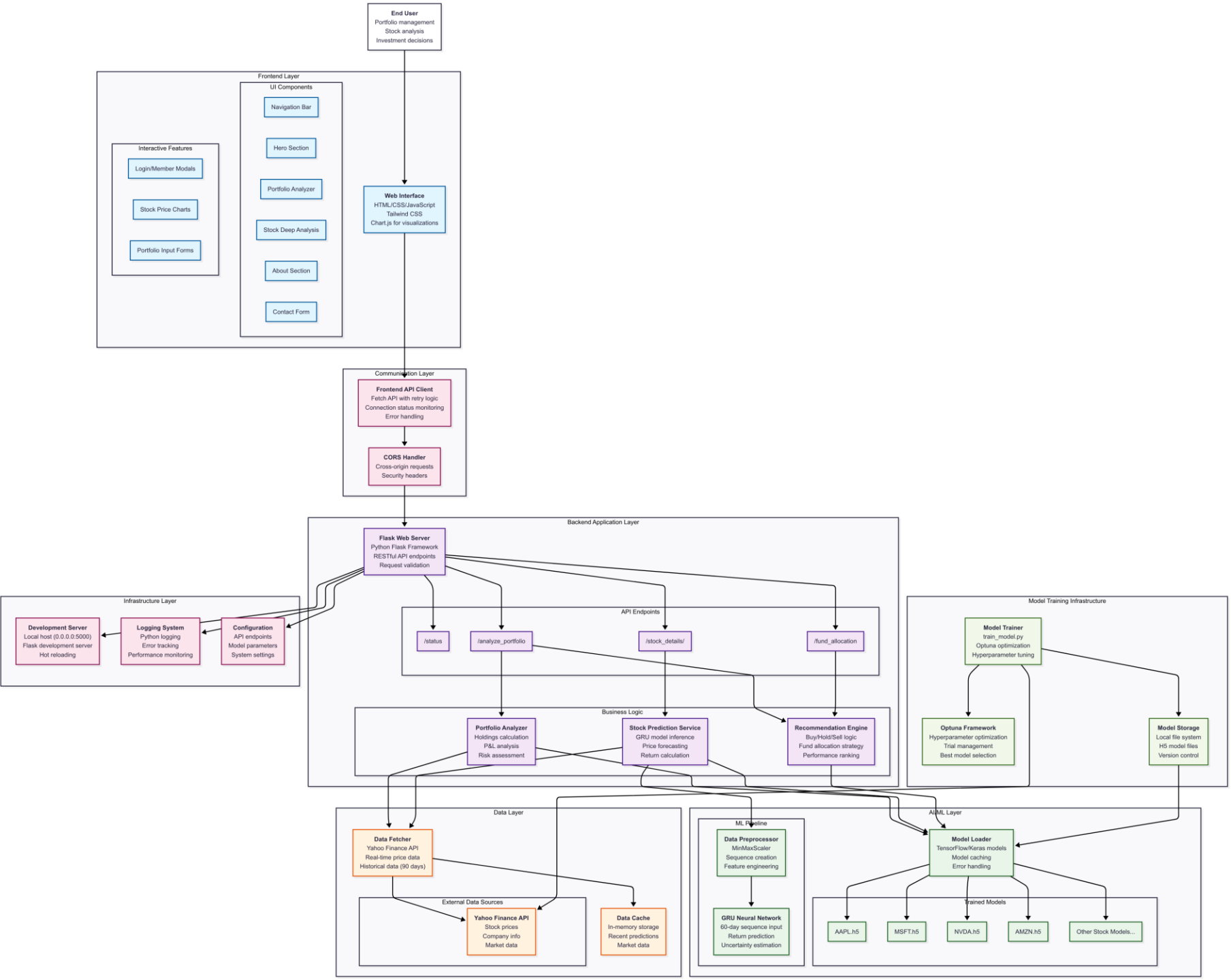
Predicted future prices; recommended stock with highest expected growth.
And predict the next 7 days stock prices.



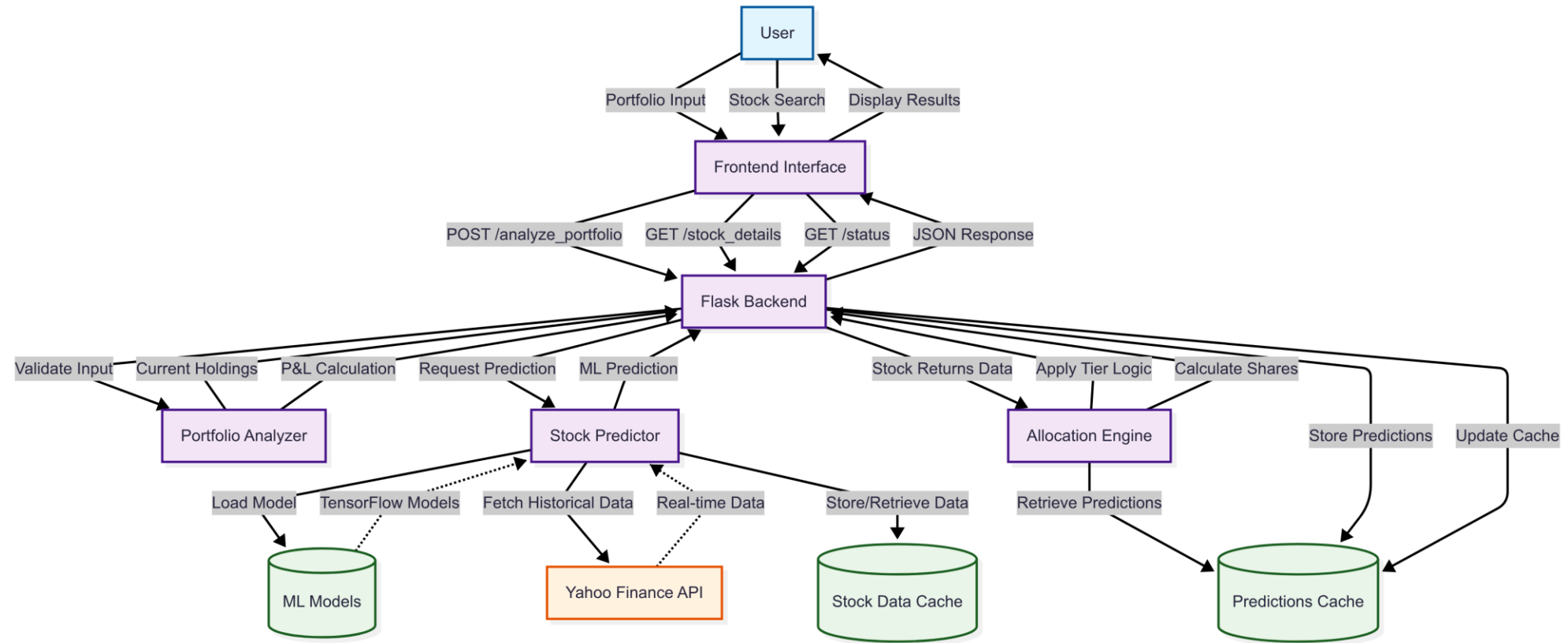
Model Architecture



System Architecture Model



Data Flow Diagram





LSTM IN STOCK MARKET PREDICTION

- LSTM is well-suited for stock price prediction as it captures time-based patterns in sequential data.
- In this project, LSTM uses the past 60 days of closing prices to predict the next 7 trading days.
- The model learns trends, fluctuations, and dependencies in price movement.
- Stacked LSTM layers improve the model's ability to detect complex patterns.
- This approach enables more accurate and consistent short-term stock forecasts compared to traditional models.



ALGORITHM FOR LSTM MODEL

1.Start

2.Load and Clean Data

- Import stock data from CSV files.
- Convert date columns and remove invalid or missing values.
- Keep only Date and Close price columns.

3.Preprocess Data

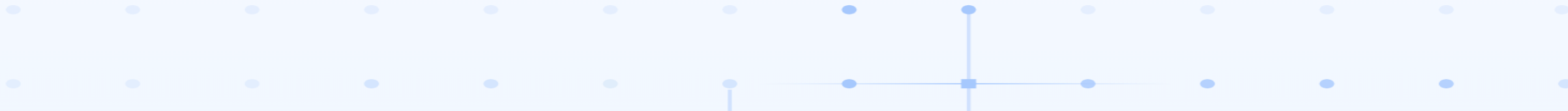
- Sort data by date.
- Normalize closing prices using MinMaxScaler.
- Create sequences of 60 consecutive prices as input and the 61st price as the output label.

4.Split Data

- Split the sequences into training and testing sets (e.g., 80%-20%).

5.Get the user input

- Enter the amount to be invested on that day.
- Enter the investment day for that initial amount.





ALGORITHM FOR LSTM MODEL (CONTD..)

6. Build LSTM model

- Initialize a Sequential model using Keras.
- Add the first LSTM layer with 50 units.
- Add the second LSTM layer with 50 units.
- Add a Dense layer with 1 unit to predict the next closing price.
- Compile the model using mean_squared_error as the loss function and adam as the optimizer.
- Fit the model on the training data for a defined number of epochs and batch size.

7. Make Predictions

- For a given investment date:
- Extract the last 60 days of prices before that date.
- Predict prices for the next 7 trading days using a rolling input window.

8. Investment Summary

- Calculate: Number of shares bought using investment amount and current price and expected profit or loss..
- Best selling day within 7-day forecast (highest predicted price).

9. End

RECOMMENDED ALLOCATION REPORT

Investment Date: 2025-04-02
Total Investment: ₹20,000.00

Stock	Return %	Allocation %	Amount	Shares	Current Price	Projected Value	Profit
TCS	1.86	50.0	₹10,000.00	2.8468	₹3,512.73	₹10,185.87	₹185.87
INFY	0.81	30.0	₹6,000.00	3.9251	₹1,528.64	₹6,048.52	₹48.52
LT	0.64	20.0	₹4,000.00	1.1805	₹3,388.30	₹4,025.56	₹25.56

Total Projected Value: ₹20,259.95
Expected Profit: ₹259.95 (1.30% ROI)



OUTPUT 2: 7 DAYS STOCK PRICE PFORECAST

- Shows 7-day predicted vs actual closing prices for TCS, INFY, and LT
- Accuracy consistently above **90%** across all 3 companies
- TCS**: Highest accuracy reached **99.37%**
- INFY**: Accuracy up to **99.49%**, showing strong prediction stability
- LT**: Slightly more variation, but still up to **98.77%**
- Predictions skip weekends to match actual market trading days
- Validates that the LSTM model performs well for short-term stock forecasting
- High accuracy supports reliable decision-making for investors

7-Day Prediction vs Actual for TCS (Skipping Weekends)			
Date	Predicted	Actual	Accuracy (%)
2025-04-03	3566.90	3373.17	94.26
2025-04-04	3513.10	3270.33	92.58
2025-04-07	3439.73	3245.21	94.01
2025-04-08	3371.75	3263.94	96.70
2025-04-09	3322.91	3218.00	96.74
2025-04-11	3276.38	3203.03	97.71
2025-04-15	3239.75	3219.58	99.37

7-Day Prediction vs Actual for INFY (Skipping Weekends)			
Date	Predicted	Actual	Accuracy (%)
2025-04-03	1531.78	1475.74	96.20
2025-04-04	1509.82	1431.51	94.53
2025-04-07	1480.38	1377.37	92.52
2025-04-08	1442.24	1408.98	97.64
2025-04-09	1419.31	1384.22	97.47
2025-04-11	1396.93	1389.89	99.49
2025-04-15	1382.96	1406.21	98.35

7-Day Prediction vs Actual for LT (Skipping Weekends)			
Date	Predicted	Actual	Accuracy (%)
2025-04-03	3500.03	3388.55	96.71
2025-04-04	3488.37	3230.03	92.00
2025-04-07	3439.95	3040.15	86.85
2025-04-08	3351.04	3131.89	93.00
2025-04-09	3293.95	3025.93	91.14
2025-04-11	3226.05	3087.16	95.50
2025-04-15	3187.86	3227.50	98.77



Dataset Acquisition

- The yfinance API is especially useful for financial projects because it simplifies
- the process of collecting clean, structured, and reliable stock market data without needing
- any API key or paid subscription. It supports a wide range of stock tickers from
- global exchanges and lets you fetch not just historical prices but also in-depth company
- insights like sector, earnings dates, dividends, stock splits

```
"C:\Users\anshu singh\AppData\Local\Programs\Python\Python313\python
YF.download() has changed argument auto_adjust default to True
[*****100%*****] 1 of 1 completed
Price      Close      High      Low      Open      Volume
Ticker      INFY.NS      INFY.NS      INFY.NS      INFY.NS      INFY.NS
Date
2017-06-30  384.817322  385.825079  382.123132  382.863517  4216700
2017-07-03  391.542511  392.817616  384.426528  387.100156  5088924
2017-07-04  395.758698  398.987608  390.514273  392.344660  5167726
2017-07-05  391.131226  394.462967  388.868925  394.462967  4226228
2017-07-06  389.527008  393.640282  388.375296  388.704353  3829720
✅ Data saved as 'infosys_7yr_data.csv'

Process finished with exit code 0
```




Gated Recurrent Unit (GRU)

Architecture

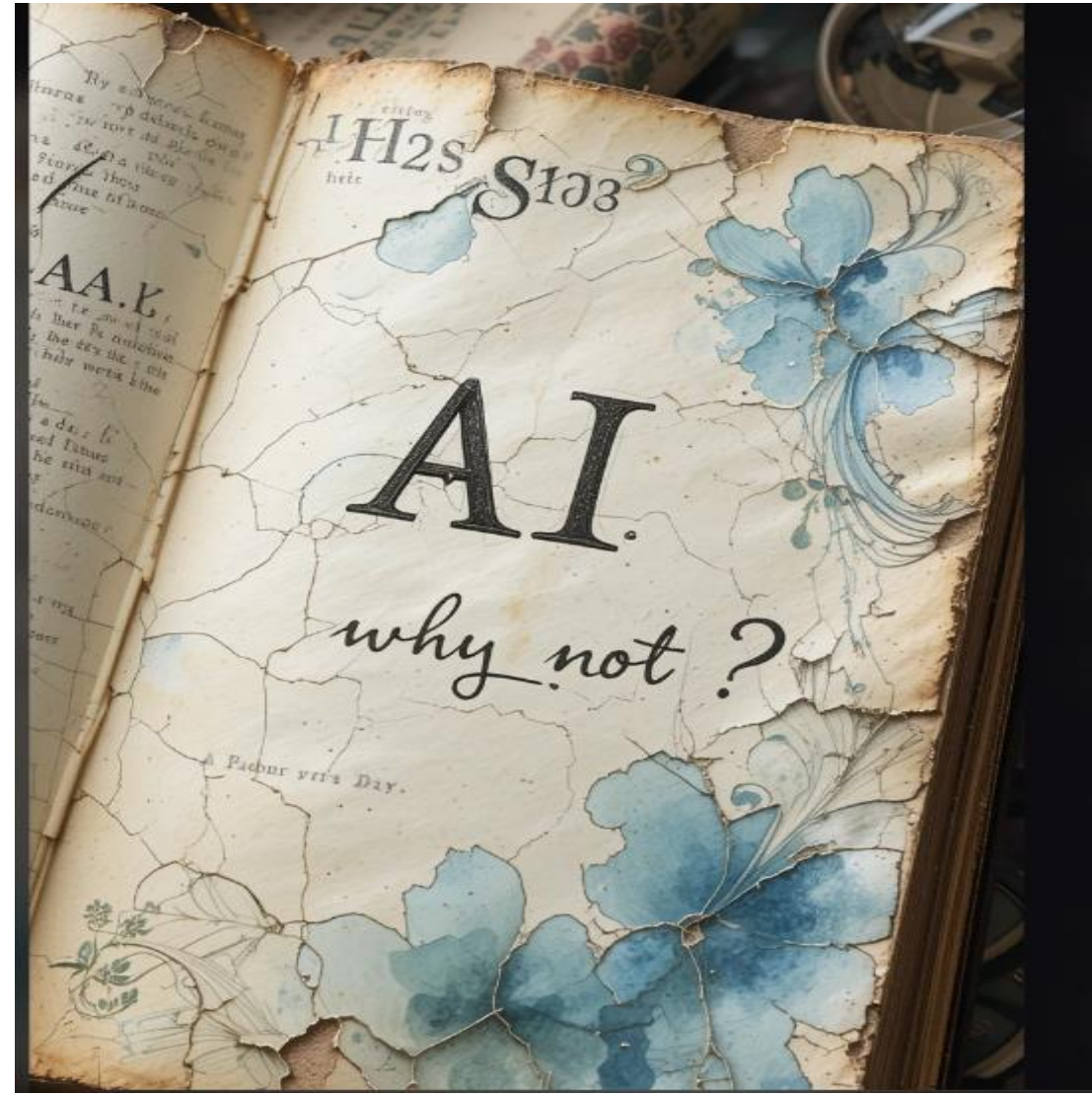
GRU is a variant of Recurrent Neural Network (RNN) specifically built for handling sequential data such as stock prices.

It is similar to LSTM but with a simpler structure and fewer gates, making it faster to train while still capturing essential temporal patterns.

Each GRU cell contains:

- **Update Gate** – determines how much of the past information to carry forward.
- **Reset Gate** – decides how much past information to forget.
- Unlike LSTM, GRU combines the forget and input gates into a single update gate, reducing complexity.

Think of GRU as a streamlined memory unit that learns what to remember and what to ignore from the





. GRU Algorithm

1. Model Construction

- Initialize a Sequential model using Keras.
- Add a GRU layer with 64 units (return_sequences=False).
- Add a Dense layer with 32 units and ReLU activation.
- Add a final Dense layer with 1 unit to predict the next closing price.
- Compile the model using mean_squared_error as the loss function and adam as the optimizer.
- Train the model on 60-day sequences of past closing prices.

2. Price Prediction

- Accept the investment date as input.
- Extract historical closing price data up to the investment date.
- Normalize the prices using MinMaxScaler.
- Create a 60-day input window ending on the investment date.
- Predict the next closing price using the trained GRU model.
- Inverse-transform the prediction to obtain the real-world price.





3. Return Estimation

- Get the actual closing price on the investment date.
- Calculate the predicted return using:
$$\frac{(\text{Predicted_Price} - \text{Current_Price})}{\text{Current_Price}} \times 100$$
- Store each stock's predicted return for comparison.

4. Portfolio Allocation

- Rank all stocks by their predicted return (descending).
- Allocate investment as follows:
 - 40% to the top performer,
 - 30% to the second-best,
 - 20% to the third,
 - 10% to the fourth.
- Exclude any stocks with negative predicted returns (allocate 0%).
- Reallocate any leftover amount to the top-performing stock.





5. Investment Summary

- For each allocated stock:
- Calculate the amount invested and number of shares bought.
- Estimate the future value using the predicted price.
- Compute expected profit or loss.
- Summarize the portfolio:
- Total investment amount,
- Total projected return,
- Net gain or loss,
- Return on Investment (ROI %).



BACKEND OUTPUT

```
hrccc.py  Version control  Current File  Run  gru_test  rc3.py  gru_test.py  nlpdem.py

STOCK ALLOCATION STRATEGY
=====
Investment Date: 2025-07-16
Total Investment: ₹20,000.00
=====

ALLOCATION BREAKDOWN:
Stock Return %  Alloc % Amount      Shares      Buy Price    Pred Price    Proj Value    Profit
-----
TCS    1.86      70.0    ₹14,000.00    4.0685      ₹3441.10      ₹3504.97      ₹14259.85      ₹259.85
INFY   0.13      30.0     ₹6,000.00     3.7313      ₹1608.00      ₹1610.03      ₹6007.59       ₹7.59
LT     -0.19      0.0      ₹0.00         0.0000      ₹3679.20      ₹3672.21      ₹0.00          ₹0.00

=====

PORTFOLIO SUMMARY
=====
Total Invested: ₹20,000.00
Total Projected Value: ₹20,267.44
Expected Profit: ₹267.44 (1.34% ROI)
=====

Process finished with exit code 0
```

pythonProject1 > gru_test.py 37:1 CRLF UTF-8 4 spaces Python 3.9 (pythonProject1)

Frontend Output

[Features](#)[Tracker](#)[About](#)[Contact](#)[Login](#)[Get Started](#)


Achieve Financial Clarity with AI

ExpenseXchange AI intelligently tracks your spending, analyzes market trends, and uncovers insights to help you build wealth and secure your financial future.

[Launch Tracker](#)

A Smarter Way to Manage Your Money


Explore the powerful features that make ExpenseXchange AI the ultimate tool for financial control.

 **ExpenseXchange AI**

Explore the powerful features that make ExpenseXchange AI the ultimate tool for financial


FeaturesTrackerAboutContact

LoginGet Started




Automated Expense Tracking

Connect your accounts or scan receipts. Our AI categorizes transactions automatically, giving you a clear view of where your money goes.




Real-Time Stock Insights

Access real-time market data, AI-driven sentiment analysis, and predictive forecasts to make smarter investment decisions.




Personalized Budgets

Create intelligent budgets that adapt to your lifestyle. ExpenseXchange AI helps you set goals and provides actionable tips to stay on track.




Portfolio Analysis

Get a holistic view of your investment portfolio. We analyze diversification, risk exposure, and performance to guide your strategy.



Bill & Subscription Tracking

Never miss a payment. ExpenseXchange AI tracks your recurring bills and subscriptions, helping you avoid late fees and cancel unused services.



Tax Optimization

Our AI identifies potential tax deductions and strategies like tax-loss harvesting to help you minimize your tax burden effortlessly.

AI Portfolio Analyzer

Available Cash (₹)

10000

Current Holdings

TCS

10

3000



LT

20

3500



+ Add Stock

Analyze Portfolio

Portfolio Summary

₹1,13,353.00

Total Value

₹10,000.00

Available Cash

2

Holdings

₹3,353.00

Total P&L

TCS 10 shares

₹30,325.00

+₹325.00 (26.75%)

LT 20 shares

₹73,028.00

+₹3,028.00 (64.43%)

All Stock Allocations

Ticker	Allocation %
RELIANCE	40.00%
LT	30.00%
TCS	20.00%
ICICIBANK	10.00%
INFY	5.00%

AI Recommendations

📈 Buy

None

— Hold

LT 1.47%
@ ₹3,645.80
TCS 1.00%
@ ₹3,030.00

📉 Sell

None

Detailed Investment Strategy

Recommended Allocations

Stock	%	Invest (₹)	Shares	CMP (₹)	Target (₹)	Profit (₹)
RELIANCE	40.0%	₹4,000.00	2.8668	₹1,395.30	₹1,415.88	₹58.99
LT	30.0%	₹3,000.00	0.8229	₹3,645.80	₹3,699.42	₹44.12
TCS	20.0%	₹2,000.00	0.6601	₹3,030.00	₹3,060.16	₹19.90
ICICIBANK	10.0%	₹1,000.00	0.6788	₹1,473.10	₹1,477.09	₹2.71
INFY	5.0%	₹500.00	0.3316	₹1,507.90	₹1,508.77	₹0.29

Total Projected Profit

₹126.01

Investment Summary

Total to Invest: ₹10,500.00

Total Stocks: 5

Remaining Cash: ₹1,02,853.00

★ Top Stocks

1	RELIANCE	₹1,395.30	1.47%	→ ₹1,415.88
2	LT	₹3,645.80	1.47%	→ ₹3,699.42
3	TCS	₹3,030.00	1.00%	→ ₹3,060.16
4	ICICIBANK	₹1,473.10	0.27%	→ ₹1,477.09
5	INFY	₹1,507.90	0.06%	→ ₹1,508.77

Top 5 Stock Forecasts

RELIANCE

₹1,395.30 → ₹1,415.88

+1.47%

LT

₹3,645.80 → ₹3,699.42

+1.47%

TCS

₹3,030.00 → ₹3,060.16

+1.00%

ICICIBANK

₹1,473.10 → ₹1,477.09

+0.27%

INFY

₹1,507.90 → ₹1,508.77

+0.06%

► [Search Stock Details](#)

INFY

Technology • Market Cap ₹6,261.69B

₹1,510.60

Current

₹1,510.90

Predicted

+0.02%

Return





Thank You

https://github.com/DankEnigmo/expensxchange_AI

