A PRELIMENERY REPORT ON

STOCK ANALYSIS (MAPAY)

SUBMITTED TO THE VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY, PUNE
IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE

OF

BACHELOR OF TECHNOLOGY (COMPUTER ENGINEERING)

SUBMITTED BY

SUDHANHU DAVE	Exam Seat No.: 22211610
REVAT MESHRAM	Exam Seat No.: 22210847

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)

BRACT'S
VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY

SURVEY NO. 3/4, KONDHWA (BUDRUK), PUNE – 411048, MAHARASHTRA (INDIA).

01. Introduction

1.1 Overview

The stock market is an ever-evolving domain where real-time insights and accurate analysis are crucial for making informed investment decisions. In today's fast-paced digital economy, the demand for intuitive, data-driven, and responsive platforms for analyzing financial markets has grown significantly.

MAPAY is a full-stack stock market analysis web application developed to address this demand, particularly for the Indian stock market. It offers users the ability to:

- Access real-time and historical data for Indian equities listed on NSE and BSE.
- Perform **technical analysis** using a wide array of indicators such as RSI, MACD, EMA, SMA, Bollinger Bands, and more.
- View categorized market movers including **Top Gainers**, **Top Losers**, and **Most Active Stocks**.
- Get **automated recommendations** based on a confidence scoring system that blends multiple indicators.

MAPAY is built with a **Flask backend** that handles data collection, processing, and analysis through yfinance and pandas_ta. The **frontend** is implemented using **React with Vite**, providing a smooth and modern user experience. To ensure fast and efficient responses, the system utilizes **asynchronous data fetching (asyncio + aiohttp)** and **in-memory caching**.

The project emphasizes:

- **Scalability**: Modular codebase and clear separation of backend/frontend for future enhancements.
- **Speed**: Asynchronous requests and caching reduce latency in fetching large amounts of financial data.
- **Accuracy**: Indicators are calculated based on standard financial formulas and validated data from Yahoo Finance.
- **User-Friendly Design**: Intuitive navigation, clear visual cues, and structured data presentation.

This system acts as both a learning tool for beginners and a practical assistant for active traders, offering insights without the complexity of traditional platforms.

1.2 Motivation

The lack of easily accessible, real-time stock analysis tools tailored for Indian markets motivated the creation of MAPAY. Existing tools often lack technical depth or usability. This project aims to bridge that gap.

1.3 Problem Definition and Objectives

- Problem: Retail investors lack tools that consolidate technical analysis, live market insights, and AI-powered predictions for Indian stocks.
- Objective: Build a full-stack system that:
    - Fetches and analyses stock data
    - Computes 12+ technical indicators
    - Offers actionable recommendations (Buy/Sell/Hold)
    - Displays top gainers/losers/most active stocks

1.4 Project Scope & Limitations

- Scope:
    - Focus on Nifty 50 stocks (major Indian indices)
    - Real-time and historical analysis
    - Simple recommendation engine
- Limitations:
    - Limited to Yahoo Finance data
    - No real trading or user portfolio integration

1.5 Methodologies of Problem Solving

- Backend: REST API using Flask + yfinance for data retrieval
- Analysis: pandas_ta for indicator calculations
- Frontend: React + Axios + Router
- Async Operations: aiohttp + asyncio for non-blocking data fetching

02. Literature Survey

Stock Market Analysis Tools: Global Landscape

Over the years, various tools have emerged to assist in stock market analysis. Platforms like **TradingView**, **MetaTrader**, and **Yahoo Finance** offer a wide range of charts, indicators, and financial data. However, these platforms are primarily designed for global or U.S.-based stocks and often lack dedicated support for Indian equities in a localized context.

Indian Context: Existing Applications

Indian investors primarily rely on platforms such as **Zerodha Kite**, **MoneyControl**, and **Groww**. These apps focus heavily on trading execution and news aggregation. While some offer basic charting tools, there is limited depth in technical analysis or advanced data science-based prediction models. None provide an integrated approach combining technical indicators with automated stock recommendations.

Technical Indicator Studies

Numerous academic papers and industry whitepapers validate the usefulness of **momentum**, **volume**, and **trend-based indicators** in predicting stock movement:

- RSI (Relative Strength Index): Popularized by J. Welles Wilder, RSI is used to identify overbought or oversold conditions.
- MACD (Moving Average Convergence Divergence): Indicates trend reversals and is widely regarded for momentum analysis.
- Bollinger Bands, Stochastic Oscillator, and ADX are commonly used for volatility and trend strength assessment. Research confirms that combining multiple indicators can improve prediction reliability, especially when backed by historical data.

Gaps Identified

- Lack of integrated platforms tailored for Indian investors combining real-time data, technical analysis, and recommendations.
- Absence of open-source solutions built on modern web stacks (Flask + React).
- Need for asynchronous, low-latency APIs that can support fast decision-making in dynamic markets.

03. System Design

The design of MAPAY follows a modular, scalable, and efficient architecture that separates concerns between frontend presentation and backend logic. The system is designed to deliver real-time stock insights with minimal latency, and the components are chosen for their lightweight nature and compatibility with asynchronous data handling.

3.1 System Architecture

MAPAY is structured into two primary layers:

- Frontend: A dynamic, single-page application built using React and Vite.
- Backend: A RESTful API built with Flask, responsible for data fetching, computation, and business logic.

Frontend Design

- Built with React (Vite) for fast performance and easy state management.
- Utilizes Axios for API communication.
- Implements React Router for page navigation between:
    - Home – Market movers (Top Gainers, Losers, Most Active)
    - Predict – Search and analyse individual stock symbols
    - Watchlist – Track user-selected stocks
    - Guide – Explanation of indicators and project logic

Backend Design

- Flask REST API exposes endpoints for:
    - /market-movers – Returns categorized market movers using Nifty 50 stocks
    - /predict – Returns detailed technical analysis for a given stock
    - /most-traded – Computes and returns stocks with the highest activity score
- Data Handling:
    - Stock data is fetched via yfinance, auto-adjusting Indian stock symbols (.NS, .BO).
    - Uses pandas_ta to compute over 12 technical indicators.
- Recommendation Logic:
    - Combines RSI, MACD histogram, and volume ratio.
    - Applies a scoring mechanism to suggest: Strong Buy, Buy, Hold, Sell, or Strong Sell.

Async and Caching System

- Implements asyncio + aiohttp for concurrent data fetching from Yahoo Finance, significantly reducing response time for endpoints handling multiple stocks.
- Caching Layer:
    - Market data is cached in-memory with a 5-minute expiration.
    - Reduces redundant API calls and improves performance.

This robust design ensures MAPAY can handle real-time data requests efficiently while being extensible for future upgrades like machine learning models, WebSocket-based live updates, and sentiment analysis integration.

04. Project Implementation

4.1 Overview of Project Modules

1. Market Movers Module

- **Endpoint**: /market-movers
- **Functionality**: Asynchronously fetches data for Nifty 50 stocks and classifies them into:
  - Top Gainers
  - Top Losers
  - Most Active (by volume)
- **Techniques**:
  - Uses asyncio and aiohttp for concurrent API requests
  - Results cached for 5 minutes to enhance performance

2. Predict Module

- **Endpoint**: /predict?symbol=RELIANCE
- **Functionality**:
  - Resolves the stock symbol (e.g., NSE/BSE)
  - Fetches real-time data via yfinance
  - Computes 12+ technical indicators using pandas_ta
  - Provides recommendation with confidence score

3. Most Traded Module

- **Endpoint**: /most-traded
- **Functionality**:
  - Calculates trading activity based on price movement and volume comparison
  - Ranks Nifty 50 stocks by activity score

4. Stock Symbol Resolver

- **Function**: try_stock_symbols()
- **Purpose**: Resolves user input to the correct Yahoo Finance symbol (e.g., TCS → TCS.NS)

5. Recommendation Engine

- Combines indicators like RSI, MACD Histogram, and Volume Ratio
- Applies scoring logic to suggest: Strong Buy, Buy, Hold, Sell, Strong Sell

4.2 Tools and Technologies Used

CATEGORY	TOOL/LIBRARY	PURPOSE
BACKEND	Flask	REST API framework
	yfinance	Fetch stock market data
	pandas_ta	Compute technical indicators
	asyncio, aiohttp	Asynchronous API requests
	logging	Debugging and tracking
FRONTEND	React (Vite)	Modern SPA frontend framework
	Axios	API calls
	React Router	Navigation between pages
OTHERS	BeautifulSoup	HTML parsing (optional for scraping)
	pandas	Data handling and manipulation
	JSON	API response formatting

4.3 Algorithm Details

4.3.1 Algorithm 1: Recommendation Engine

Inputs: RSI, MACD Histogram, Volume Ratio
Steps:

1. Assign points based on RSI:
    - RSI < 30 → +2 (Strong Buy)
    - RSI < 40 → +1 (Buy)
    - RSI > 70 → +2 (Strong Sell)
    - RSI > 60 → +1 (Sell)
2. Adjust based on MACD Histogram:
    - Positive → reinforce Buy signal
    - Negative → reinforce Sell signal
3. Volume Ratio > 1.5 → +1 confidence (confirms momentum)
4. Final Recommendation:
    - Score ≥ 3 → "Strong Buy" / "Strong Sell"
    - Score 1–2 → "Buy" / "Sell"
    - Else → "Hold"

4.3.2 Algorithm 2: Trading Activity Scoring

Inputs: Current volume, previous volume, price change
Steps:

1. Calculate volume_ratio = current_volume / previous_volume
2. Calculate price_change = ((current - prev) / prev) * 100
3. Score = volume_ratio * (1 + |price_change / 100|)
4. Rank all stocks by this score to determine most traded stocks

4.3.3 Algorithm 3: Market Movers Classification

Inputs: Price change %, Volume
Steps:

1. For each stock, compute:
    - price_change = ((today - yesterday) / yesterday) * 100
2. Sort:
    - Descending → Top Gainers
    - Ascending → Top Losers
    - By Volume → Most Active

This structured implementation ensures the system delivers actionable insights in near real-time while remaining scalable and efficient. All modules are loosely coupled, enabling future enhancements like deep learning integration or sentiment-based analysis.

5.1 Outcomes

Real-Time Market Data Retrieval
MAPAY successfully retrieves up-to-date stock data for Nifty 50 stocks using Yahoo Finance via the yfinance library. The data includes open, close, high, low, volume, and historical trends.
Accurate Technical Indicator Calculation
The system computes 12+ technical indicators such as:

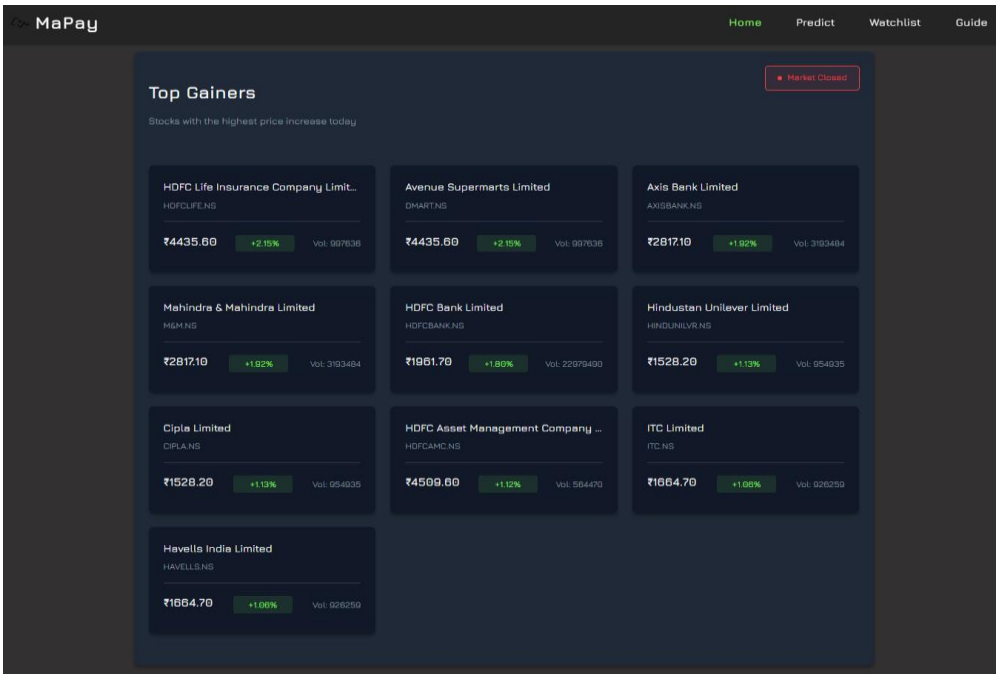
- RSI, MACD, EMA, SMA
- Bollinger Bands
- VWAP, OBV
- ADX, Stochastic Oscillator, Volume Ratio

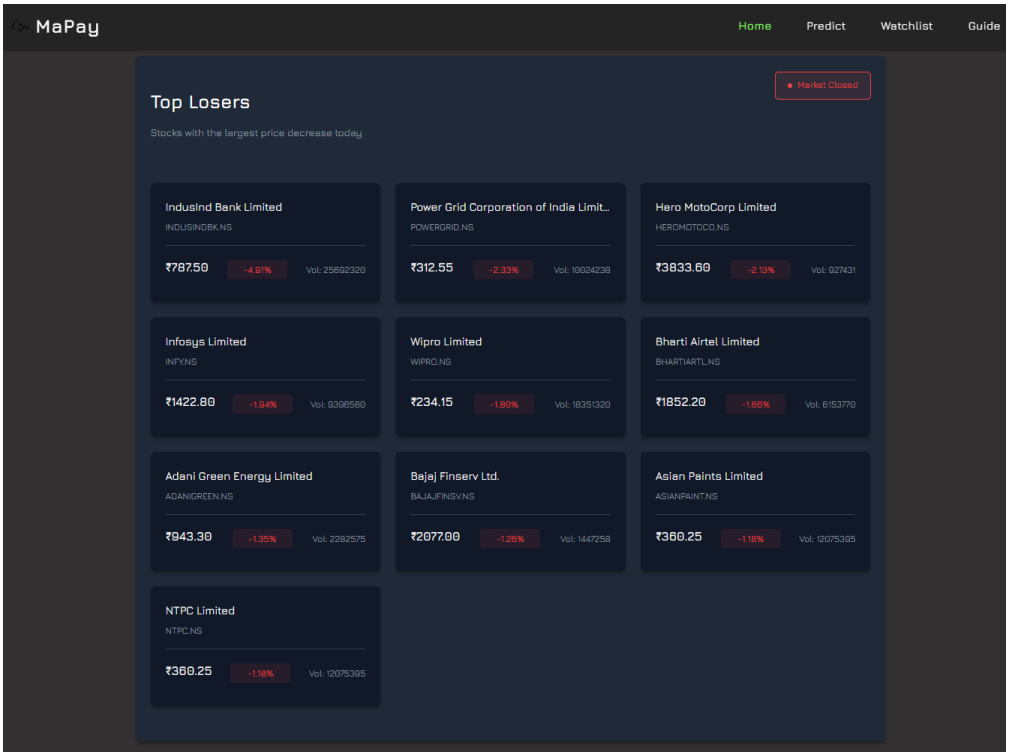
These indicators are used to provide reliable trading signals.
Functional REST API
All backend endpoints (/market-movers, /predict, /most-traded) return consistent and structured JSON responses. API latency is optimized using asynchronous fetching and caching.
Recommendation Engine
The scoring-based recommendation engine produces Buy/Sell/Hold suggestions with confidence levels. Users can interpret these based on technical indicator signals, enhanced by volume confirmation.
Frontend Integration
The React-based frontend dynamically consumes data from the backend and provides a smooth navigation experience across:

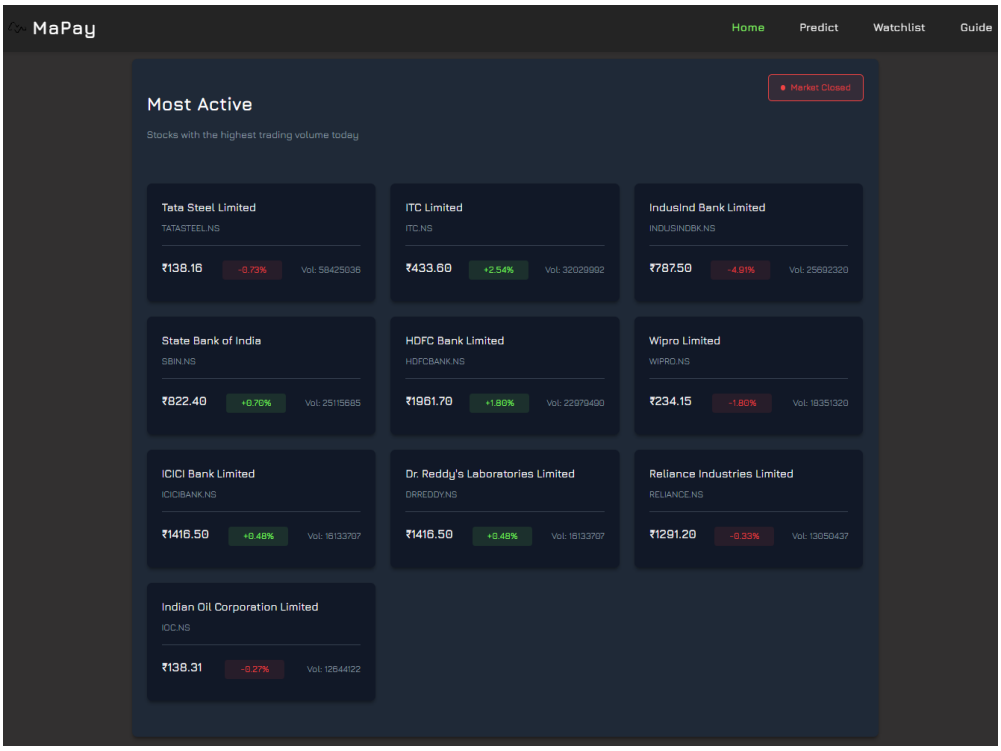
- Home (Top Movers)
- Predict (Detailed Analysis)
- Watchlist

5.2 Screenshots

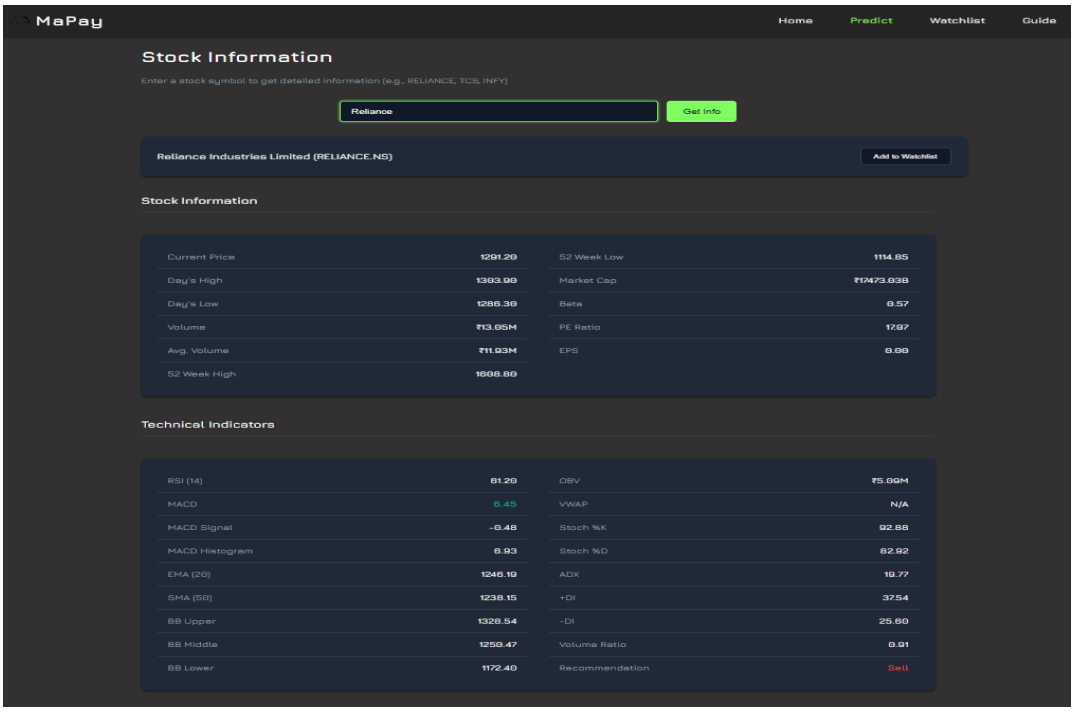
- **Home Page – Top Gainers, Losers, Most Active Stocks**



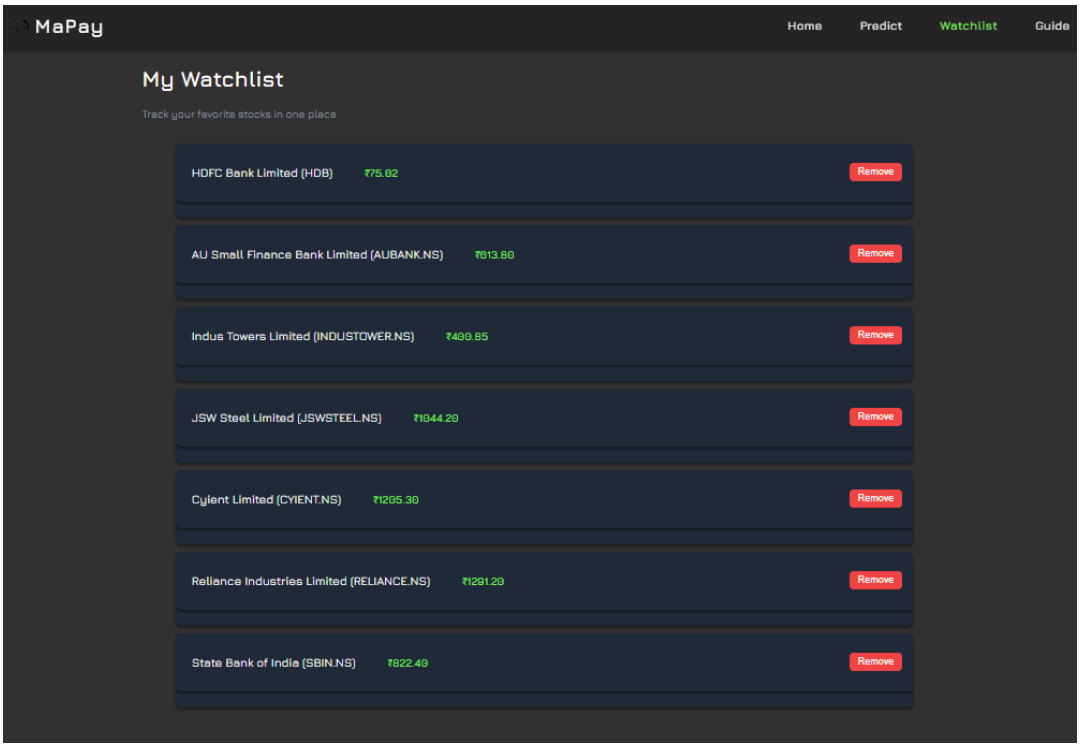

MaPay
Home
Predict
Watchlist
Guide
Market Closed
Top Losers
Stocks with the largest price decrease today
IndusInd Bank Limited
INDUSINDBK.NS
₹787.50
-4.91%
Vol: 25892320
Power Grid Corporation of India Limit...
POWERGRID.NS
₹312.55
-2.33%
Vol: 19824238
Hero MotoCorp Limited
HEROMOTOCO.NS
₹3833.60
-2.13%
Vol: 927431
Infosys Limited
INFY.NS
₹1422.80
-1.94%
Vol: 9396560
Wipro Limited
WIPRO.NS
₹234.15
-1.80%
Vol: 18351320
Bharti Airtel Limited
BHARTIARTL.NS
₹1852.20
-1.66%
Vol: 6153770
Adani Green Energy Limited
ADANIGREEN.NS
₹943.30
-1.35%
Vol: 2282575
Bajaj Finserv Ltd.
BAJAJFINSV.NS
₹2077.00
-1.26%
Vol: 1447258
Asian Paints Limited
ASIANPAINT.NS
₹360.25
-1.18%
Vol: 12075395
NTPC Limited
NTPC.NS
₹360.25
-1.18%
Vol: 12075395


MaPay
Home
Predict
Watchlist
Guide
Market Closed
Most Active
Stocks with the highest trading volume today
Tata Steel Limited
TATASTEEL.NS
₹138.16
-0.73%
Vol: 58425036
ITC Limited
ITC.NS
₹433.60
+2.54%
Vol: 32029992
IndusInd Bank Limited
INDUSINDBK.NS
₹787.50
-4.91%
Vol: 25892320
State Bank of India
SBIN.NS
₹822.40
+0.76%
Vol: 25115685
HDFC Bank Limited
HDFCBANK.NS
₹1961.70
+1.80%
Vol: 22979490
Wipro Limited
WIPRO.NS
₹234.15
-1.80%
Vol: 18351320
ICICI Bank Limited
ICICIBANK.NS
₹1416.50
+0.48%
Vol: 16133707
Dr. Reddy's Laboratories Limited
DRREDDY.NS
₹1416.50
+0.48%
Vol: 16133707
Reliance Industries Limited
RELIANCE.NS
₹1291.20
-0.33%
Vol: 13050437
Indian Oil Corporation Limited
IOC.NS
₹138.31
-0.27%
Vol: 12644122

- **Predict Page – Search results with stock data, indicator values, and recommendation**



- **Watchlist – Display of saved stock symbols**



06. Conclusions

6.1 Conclusions

MAPAY successfully delivers a fast, reliable, and intuitive platform for real-time stock market analysis, tailored specifically for the Indian stock market (NSE/BSE). It bridges the gap between complex trading platforms and simplified mobile investment apps by offering a middle ground—focused on technical analysis, automated recommendations, and real-time market insights.

- Backend efficiency: Leveraged Flask, yfinance, and pandas_ta to fetch and analyze stock data dynamically.
- Asynchronous data handling: Implemented asyncio and aiohttp to fetch multiple stock records simultaneously with minimal delay.
- User-friendly frontend: Developed using React and Vite, offering a modern UI/UX that's responsive and informative.
- Modular architecture: Separation of concerns between backend APIs and frontend interfaces ensures easy scalability and maintenance.
- Intelligent recommendation engine: Provided buy/sell/hold signals based on scoring logic using momentum (RSI), trend (MACD), and volume confirmation.

In summary, MAPAY serves as a highly functional prototype of a stock analysis tool, demonstrating how modern web technologies can empower investors with insights that were traditionally available only on premium platforms.

6.2 Future Work

While MAPAY delivers on its initial goals, several enhancements can further elevate its capabilities:

- Integrate Machine Learning:
    - Use deep learning models like LSTM, Informer, or Autoformer for more accurate price predictions.
    - Implement back testing frameworks to validate prediction performance over time.
- Real-Time Data Streaming:
    - Add WebSocket or Server-Sent Events (SSE) to push live stock prices and indicator updates to users without refreshing.
- Sentiment Analysis Integration:
    - Combine financial news and social media sentiment using NLP to improve recommendations and flag events like earnings reports or market crashes.

6.3 Applications

MAPAY has practical use cases in various domains:

- Retail Trading: Aids traders in making data-driven decisions using live indicators and recommendations.
- Financial Education: Acts as a learning tool for students and beginners to understand the role of technical indicators.
- Research & Academia: Can be used to simulate trading strategies or validate financial theories.
- Fintech Prototyping: Provides a ready framework for startups looking to build stock-related products.

References

1. Thomas Noltey, Hans Hanssony, Lucia Lo Belloz, "**Communication Buses for Automotive Applications**", Proceedings of the 3rd Information Survivability Workshop (ISW-2007), Boston, Massachusetts, USA, IEEE Computer Society.
2. yfinance Python Library - https://github.com/ranaroussi/yfinance
3. pandas_ta Technical Analysis Library - https://github.com/twopirllc/pandas-ta
4. Flask Web Framework - https://flask.palletsprojects.com/
5. ReactJS Library - https://react.dev/
6. "Technical Analysis of the Financial Markets" by John J. Murphy (1999)
7. Investopedia - Technical Indicators section: https://www.investopedia.com/technical-analysis-4689746
8. Zerodha Varsity - https://zerodha.com/varsity/