R. M. K. COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution)

R.S.M Nagar, Puduvoyal - 601206.

NEWS RESEARCH TOOL FOR EQUITY ANALYSIS USING LANGCHAIN

A PROJECT REPORT

Submitted by

ALAPATI DEVI ANUSHA	**(111620243003)**
BATTEPATI DHARANI	**(111620243006)**
PUNURU DIVYA HARSHITHA	**(111620243041)**
VADLAMUDI LAKSHMI TEJASWINI	**(111620243056)**

BACHELOR OF TECHNOLOGY

IN

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

R.M.K. COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution)
R.S.M. Nagar, Puduvoyal 601206

ANNA UNIVERSITY, CHENNAI 600 025
MARCH 2024

BONAFIDE CERTIFICATE

Certified that this project report **“NEWS RESEARCH TOOL FOR EQUITY ANALYSIS USING LANGCHAIN”** is the bonafide work of **ALAPATI DEVI ANUSHA (111620243003), BATTEPATI DHARANI (111620243006), PUNURU DIVYA HARSHITHA (111620243041) and VADLAMUDI LAKSHMI TEJASWINI (111620243056)** who carried out the project work under my supervision.

SIGNATURE

Dr.B. Prathusha Laxmi,
Professor and Head,

Department of AI & DS,

R.M.K College of Engineering and Technology, Puduvoyal.

SIGNATURE

Dr. P. Josephin Shermila,
Associate Professor and Supervisor,

Department of AI & DS,

R.M.K College of Engineering and Technology, Puduvoyal.

Certified that the above candidate was examined in the university project viva-voce held on___________________.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

A project of this magnitude and nature requires kind co-operation and support from many, for successful completion. We wish to express our sincere thanks to all those who were involved in the completion of this project.

Our sincere thanks to Our Honorable Founder and Chairman **Vidya Ratna Thiru. R.S.Munirathinam, Ex. MLA** for his sincere endeavor in educating us in his premier institution.

We would like to express our deep gratitude to Our Beloved Vice Chairman, **Thiru. R.M.Kishore,** for his kind words and enthusiastic motivation which inspired us a lot in completing this project.

We also express our appreciation and gratefulness to Our Principal **Dr.K.Ramar** and Our sincere thanks to **Dr.K.Sivaram**, Dean (Research) for giving support to complete the project.

We wish to convey our thanks and gratitude to **Dr.B.Prathusha Laxmi,** Professor and Head of Artificial Intelligence and Data Science Department, for her support and by providing us ample time to complete our project.

We express our indebtedness and gratitude to our Project Supervisor, **Dr.P.Josephin Shermila,** Associate Professor**,** Department of Artificial Intelligence and Data Science for her guidance throughout the course of our project.

Last, but not the least, we take this opportunity to thank all the faculty members and supporting staff of the Department of Artificial Intelligence and Data Science.

TABLE OF CONTENTS

ABSTRACT

The OpenAI-driven Equity Analysis Platform is a web-based tool designed to transform traditional equity research. The proposed model utilizes advanced natural language processing. It enables users to input up to four recent links related to a specific company's market and stock trends. It presents the development of an innovative Equity Research News Tool leveraging Langchain, OpenAI, and Facebook AI Similarity Search (FAISS). Langchain facilitates natural language processing tasks, enabling efficient extraction and summarization of relevant news articles. OpenAI enhances language understanding and generation capabilities, while FAISS optimizes similarity search for efficient data retrieval. Integrating these technologies empowers analysts to streamline equity research processes, extract actionable insights, and make informed investment decisions in real-time.

The system utilizes OpenAI's language models to provide accessible and efficient equity analysis through a user-friendly interface with question-answering capabilities. Design and implement a responsive, user-friendly interface, prioritizing accessibility to provide a streamlined and efficient experience for investors and research analysts alike. Unlike existing models requiring manual copy-pasting of entire articles, our approach simplifies the process. Analysts only need to paste the links to recent articles about a specific company.

The model autonomously accesses the provided links, analyzes the entire articles using advanced technologies like Langchain for NLP tasks and the OpenAI API for text generation.

CHAPTER 1

INTRODUCTION

This chapter gives introduction to the problem, also includes the information about the existing system and their disadvantages, proposed system and its advantages along with future enhancement. The Equity Research News Tool is a comprehensive language model (LLM) project designed for equity research analysts. Leveraging the power of langchain, the Open AI API, and FAISS, this tool allows analysts to conduct efficient and insightful news research in the ever-evolving financial landscape, it uses fancy language models to understand recent news about a company's stocks and market trends. Just share a few recent links, and the tool will give you insights without the jargon. Our goal is to simplify trading by using cool technology from OpenAI to navigate the complex world of finance.

Traditional equity analysis struggles with manually gathering and analyzing vast amounts of news data, leading to inefficiency and potentially missing crucial information. This Langchain-powered tool aims to automate news collection, utilize AI for sentiment analysis, and deliver actionable insights for informed investment decision. The primary objective of this project is to create a user-friendly web application that enables equity research analysts to gather relevant news articles based on their queries, perform natural language processing (NLP) tasks using langchain, generate human-like text with OpenAI, and utilize FAISS for efficient similarity search.

In the past, our groundbreaking equity analysis model introduces a significant departure from conventional methods. Unlike existing models requiring manual copy- pasting of entire articles, our approach simplifies the process. Analysts only need to paste the links to recent articles about a specific company.

The model autonomously accesses the provided links, analyzes the entire articles using advanced technologies like Langchain for NLP tasks and the OpenAI API for text generation. This innovation not only enhances efficiency by eliminating manual input but also offers a more streamlined experience for research analysts.

The use of the project of building an Equity Research News Tool with Langchain, OpenAI, and FAISS serves various purposes and applications such as Market Analysis, Investment Decision, Risk Management, Real-time Updates. The tool can be utilized by financial analysts and researchers to gather, analyze, and summarize news articles related to equities, enabling them to stay informed about market trends, company developments, and industry news. Investment Decision making for Investors can use the tool to access curated news content and extract valuable insights, helping them make informed decisions regarding stock purchases, portfolio management, and trading strategies. Risk Management monitors news sentiment and identifies potential risks or opportunities. Risk managers can use the tool to assess and mitigate risks associated with equity investments. Real-time Updates The tool provides real-time updates on market news and events, ensuring that users have access to the latest information and can react promptly to changes in the market environment.

Overall, the Equity Research News Tool offers a comprehensive solution for accessing, analyzing, and interpreting news content related to equities, empowering users with valuable insights to support their investment decisions and strategic planning efforts. collected from the individual’s medical history and physical examination and from other tests performed.

Furthermore, the integration of OpenAI augments the tool's language understanding and generation capabilities. By leveraging state-of-the-art natural language processing models, our tool can interpret complex financial terminology, infer context, and extract nuanced insights from news articles. This enhanced comprehension enables analysts to gain deeper insights in to market trends, company performance, and industry dynamics.

Additionally, FAISS enhances the tool's data retrieval efficiency by optimizing similarity search algorithms. With FAISS, analysts can quickly locate and access relevant articles based on similarity metrics, enabling them to conduct comprehensive research and make well- informed investment decisions in real-time.

In the fast-paced world of equity research, staying abreast of the latest news and developments is paramount for informed decision-making. However, the sheer volume and complexity of available information pose significant challenges for analysts. To address this, we present an innovative solution leveraging cutting-edge technologies: Langchain, OpenAI, and FAISS. By combining the power of natural language processing, advanced language understanding, and efficient data retrieval, our Equity Research News Tool revolutionizes the way analysts gather, analyze, and extract insights from news articles related to equities.

Unlike the existing model, the proposed system requires manual copy-pasting of entire articles which simplifies the process. It provides Analysts only to paste the links of recent articles about a specific company. The model autonomously accesses the provided links, analyzes the entire articles using advanced technologies like Langchain for NLP tasks and the OpenAI API for text generation.

The proposed system reduces the analysis time needed for the analyst especially when there are multiple articles, and in cases of large numbers articles to be analysed.

The primary objective of this project is to create a user-friendly web application that enables equity research analysts to gather relevant news articles based on their queries, perform natural language processing (NLP) tasks using langchain, generate human-like text with OpenAI, and utilize FAISS for efficient similarity search.

Traditional equity analysis struggles with manually gathering and analyzing vast amounts of news data, leading to inefficiency and potentially missing crucial information. This Langchain-powered tool aims to automate news collection, utilize AI for sentiment analysis, and deliver actionable insights for informed investment decision.

1.1 System Study

Invest IT Bot aims to revolutionize how users engage with financial news and research, providing a streamlined platform for accessing relevant information. The feasibility study assesses the viability of this endeavor from various perspectives.

A thorough market analysis reveals the dynamic landscape of financial technology (fintech) and the growing demand for innovative tools in investment research. By analyzing market trends, competitor offerings, and user preferences, we gain insights in to the potential market share and positioning of Invest IT Bot.

Invest IT Bot relies on sophisticated technologies such as natural language processing (NLP), machine learning, and web scraping to gather and analyze news articles. The technical feasibility study evaluates the scalability, reliability, and compatibility of these technologies to ensure seamless operation and user experience.

Cost estimation involves assessing development expenses, infrastructure costs, and ongoing maintenance requirements. Revenue projections consider subscription models, advertisement revenue, and potential partnerships, balancing investment with expected returns to determine financial viability.

Navigating legal and regulatory frameworks is critical in the fintech sector. Compliance with data protection laws, financial regulations, and intellectual property rights is essential to avoid legal hurdles and build trust among users and stakeholders.

Identifying potential risks, such as data breaches, market volatility, or technological disruptions, enables proactive risk mitigation strategies. By quantifying risks and their potential impact, Invest IT Bot can implement contingency plans to minimize adverse effects on operations and reputation.

Based on the findings of the feasibility study, we conclude that Invest IT Bot holds significant promise in addressing the growing demand for efficient financial research tools. However, careful consideration of market dynamics, technical requirements, financial implications, and regulatory compliance is necessary to ensure successful implementation and sustainable growth.

Three key considerations involved in the feasibility analysis are

- ECONOMIC FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

1.1.1 Economic Feasibility:

Economic Impact:

Invest IT Bot's launch is expected to stimulate economic activity by enhancing market efficiency and facilitating informed investment decisions. By providing users with timely, accurate information, the application contributes to improved asset allocation and resource allocation in the financial markets.

Cost-Benefit Analysis:

A comprehensive cost-benefit analysis evaluates the financial implications of developing and operating Invest IT Bot. While initial investments may be substantial, the potential long-term benefits, such as revenue generation, market expansion, and user satisfaction, justify the expenditure.

Market Growth Potential:

Invest IT Bot's innovative features and user-centric design position it as a catalyst for market growth. By attracting new users and expanding market reach, the application stimulates demand for financial services and fosters innovation within the fintech ecosystem.

Employment Impact:

The deployment of Invest IT Bot creates job opportunities in software development, data analysis, customer support, and marketing. Additionally, the application's broader economic impact stimulates job growth in related sectors, contributing to overall employment stability and prosperity.

Economic Sustainability:

Sustainable economic growth hinges on Invest IT Bot's ability to adapt to evolving market dynamics and user needs. Continuous innovation, strategic partnerships, and prudent financial management are essential for maintaining economic sustainability and maximizing long-term value creation.

1.1.2 Technical Feasibility

Technology Stack:

Invest IT Bot leverages cutting-edge technologies, including natural language processing (NLP), machine learning algorithms, and cloud computing infrastructure. The technical study evaluates the robustness, scalability, and security of these technologies to ensure optimal performance and user satisfaction.

Scalability Analysis:

Scalability is critical to accommodate growing user demand and data volume.By assessing scalability metrics, such as response time, throughput, and resource utilization, Invest IT Bot identifies scalability bottlenecks and implements scalable solutions to ensure seamless scalability.

Security Analysis:

Data security is paramount in financial applications. Invest IT Bot employs encryption, access controls, and intrusion detection systems to safeguard user data and transactions. Regular security audits and penetration testing ensure compliance with industry standards and regulatory requirements.

System Architecture:

Invest IT Bot's system architecture comprises interconnected modules for data ingestion, processing, analysis, and presentation. The study examines system components, interfaces, and dependencies to optimize performance, reliability, and maintainability.

Maintenance and Support:

Effective maintenance and support are essential for ensuring Invest IT Bot's continued operation and user satisfaction. A robust maintenance strategy, including software updates, bug fixes, and user support services, enhances application reliability and longevity.

1.1.3 Social Feasibility

User Impact Analysis:

Invest IT Bot's user-centric design aims to empower users with access to comprehensive financial information and insights. User impact analysis evaluates user satisfaction, engagement, and behavior changes to measure the application's effectiveness in meeting user needs.

Accessibility and Inclusivity:

Invest IT Bot prioritizes accessibility features to ensure inclusivity for users with diverse needs and abilities. Compliance with accessibility standards, such as WCAG guidelines, enables users with disabilities to access and benefit from the application's features.

Ethical Considerations:

Ethical considerations guide Invest IT Bot's development and operation, ensuring transparency, fairness, and integrity in financial research and analysis. Ethical guidelines govern data privacy, content moderation, and algorithmic fairness to promote ethical behavior and user trust.

Community Engagement:

Invest IT Bot fosters community engagement through feedback mechanisms, user forums, and educational resources. By soliciting user feedback and addressing community concerns, the application cultivates a supportive and collaborative user community.

Social Responsibility:

As a socially responsible platform, Invest IT Bot promotes financial literacy, empowerment, and inclusion. Educational initiatives, philanthropic partnerships, and community outreach programs underscore the application's commitment to social impact and responsible citizenship.

1.2 Existing System

The existing system model for artificial intelligence applied to stock market trading encompasses a wide array of research and practical applications. Over the years, numerous studies have delved into the utilization of machine learning and deep learning algorithms to predict stock market movements, optimize portfolios, and analyze financial sentiment. A systematic review of literature spanning from the 1990s to recent years reveals the evolution and increasing specificity within this domain.

Beginning with foundational research in portfolio optimization, scholars have explored various approaches aimed at maximizing returns while minimizing risks. Techniques such as the Holt- Winters algorithm and neural networks have been incorporated to forecast future stock prices based on historical data. This integration of time series forecasting systems with advanced algorithms has provided investors with tools to make informed decisions in volatile market conditions.

Moreover, the convergence of machine learning and deep learning methodologies has revolutionized stock market prediction. Recent studies have demonstrated the efficacy of ensemble models combining algorithms like Random Forest, XG-Boost, and Long Short-Term Memory (LSTM) networks. These models offer a comprehensive framework for forecasting stock prices and classifying investment opportunities, thereby empowering traders with actionable insights.

One notable aspect of the existing system model is its adaptation to real-time data streams from diverse sources such as stock exchanges, economic reviews, and social media platforms. Machine learning algorithms leverage this influx of data to extract meaningful

patterns and trends, incorporating technical indicators like rolling mean and momentum to enhance predictive accuracy. Additionally, the incorporation of qualitative factors such as news sentiment analysis further enriches the predictive capabilities of these models.

The existing system model thus represents a sophisticated ecosystem where artificial intelligence intersects with financial markets, driving innovation and efficiency in investment strategies. By harnessing the power of machine learning and deep learning, investors can navigate the complexities of the stock market with greater confidence and agility, ultimately leading to more informed and profitable decision-making.

The following are some of the disadvantages of the existing system :

Data Dependency and Quality: The accuracy and reliability of AI-based stock market predictions heavily rely on the quality and quantity of data available. Inaccurate or incomplete data inputs can lead to flawed predictions and investment decisions, potentially resulting in financial losses.

Overfitting and Generalization Issues: Machine learning models trained on historical data may suffer from overfitting, where they learn to memorize the training data rather than generalize patterns. This can lead to poor performance when applied to new, unseen data, undermining the effectiveness of the predictive model.

Market Volatility and Unforeseen Events: AI models may struggle to adapt to sudden changes or unexpected events in the stock market, such as economic crises, geopolitical tensions, or natural disasters. These unforeseen circumstances can disrupt typical market patterns and renderAI-based predictions inaccurate or unreliable.

Algorithmic Bias and Interpretability: Machine learning algorithms can exhibit biases based on the data they are trained on, potentially leading to skewed predictions or discriminatory outcomes. Moreover, the inner workings of complex AI models like neural networks may lack interpretability, making it difficult for investors to understand the rationale behind specific predictions.

Cybersecurity Risks: As AI systems become increasingly integrated into financial markets, they become potential targets for cyberattacks and exploitation. Malicious actors could manipulate AI algorithms or compromise data integrity, leading to fraudulent activities or market manipulation.

1.3 Proposed System

The proposed system, the "Invest IT Bot: News Research Tool 📈," offers several advancements over the existing system model for applying artificial intelligence to stock market trading. Here's how the proposed system is superior:

Interactive User Experience: With Streamlit, the proposed system provides a more interactive and intuitive user experience compared to traditional command-line interfaces. Users can input news article URLs directly through the graphical user interface, eliminating the need for manual data preprocessing steps. This streamlined approach enhances user engagement and efficiency.

Advanced Natural Language Processing: The integration of OpenAI's language model (LLM) enables more sophisticated natural language understanding and generation capabilities. The system can interpret user queries in natural language and provide comprehensive answers sourced from relevant articles. Additionally, the RetrievalQA With Sources Chain allows for contextualized responses by considering multiple sources simultaneously, offering deeper insights into queried topics.

Efficient Information Retrieval: The use of FAISS for indexing and searching embedded document vectors enhances the speed and efficiency of information retrieval compared to traditional search methods. FAISS's ability to perform fast similarity searches enables quick access to relevant articles, empowering users to make informed decisions in real-time trading scenarios.

Scalability and Adaptability: The modular architecture of the proposed system facilitates scalability and adaptability to evolving market trends and user requirements. New features, such as sentiment analysis, sentiment tracking, or sentiment-driven trading strategies, can be seamlessly integrated into the system to enhance its functionality and relevance over time.

In summary, the Invest IT Bot represents a significant advancement in the field of AI-driven stock market analysis and decision-making tools. By leveraging real-time data processing, interactive user interfaces, advanced natural language processing, efficient information retrieval, and scalable architecture, the proposed system offers a superior alternative to traditional approaches, empowering users with timely and actionable insights for successful trading in dynamic financial markets.

CHAPTER 2
LITERATURE SURVEY

The application of Artificial Intelligence (AI) to financial investment is a research area that has attracted extensive research attention since the 1990s, when there was an accelerated technological development and popularization of the personal computer. Since then, countless approaches have been proposed to deal with the problem of price prediction in the stock market. This presents a systematic review of the literature on Artificial Intelligence applied to investments in the stock market based on a sample of 2326 papers from the Scopus website between 1995 and 2019.These papers were divided into four categories: portfolio optimization, stock market prediction using AI, financial sentiment analysis, and combinations involving two or more approaches. Foreach category, the initial introductory research to its state-of-the-art applications are described. In addition, an overview of the review leads to the conclusion that this research area is gaining continuous attention and the literature is becoming increasingly specific and thorough.

2.1 A Novel AI-Based Stock Market Prediction Using Machine Learning Algorithm:

The time series forecasting system can be used for investments in a safe environment with minimized chances of loss. The Holt–Winters algorithm followed various procedures and observed the multiple factors applied to the neural network. The final module helps filter the system to predict the various factors and provides a rating for the system. This research work uses real-time dataset of fifteen stocks as input into the system and, based on the data, predicts or forecasts future stock prices of different companies belonging to different sectors. The dataset includes approximately fifteen companies from different sectors and forecasts their results based on which the user can decide whether to invest in the particular company or not the forecasting will give an accurate result for the customer investments.

2.2 Forecasting Stock Market Prices Using Machine Learning and Deep Learning Models: A Systematic Review, Performance Analysis and Discussion of Implications:

The financial sector has greatly impacted the monetary well-being of consumers, traders, and financial institutions. In the current era, artificial intelligence is redefining the limits of the financial markets based on state-of-the-art machine learning and deep learning algorithms. There is extensive use of these techniques in financial instrument price prediction, market trend analysis, establishing investment opportunities, portfolio optimization, etc. Investors and traders are using machine learning and deep learning models for forecasting financial instrument movements. With the widespread adoption of AI in finance, it is imperative to summarize the recent machine learning and deep learning models, which motivated us to present this comprehensive review of the practical applications of machine learning in the financial industry. This article examines algorithms such as supervised and unsupervised machine learning algorithms, ensemble algorithms, time series analysis algorithms, and deep learning algorithms for stock price prediction and solving classification problems. The contributions of this review article are as follows: (a) it provides a description of machine learning and deep learning models used in the financial sector; (b) it provides a generic framework for stock price prediction and classification; and (c) it implements an ensemble model—"Random Forest + XG- Boost + LSTM"—for forecasting TAINIWALCHM and AGROPHOS stock prices and performs a comparative analysis with popular machine learning and deep learning models.

2.3 Algorithmic Machine Learning for Prediction of Stock Prices:

Stock markets and relevant entities generate enormous amounts of data on a daily basis and are accessible from various channels such as stock exchange, economic reviews, and employer monetary reports. In recent times, machine learning techniques have proven to be very helpful in making better trading decisions. Machine learning algorithms use complex logic to observe and learn the behavior of stocks using historical data which can be used to predict future movements of the stock. Technical indicators such as rolling mean, momentum, and exponential moving average are calculated to convert the data into meaningful information. Furthermore, this information can be used to build machine learning prediction models that learn different patterns in the data and make future predictions for accurate financial forecasting. Additional factors that are being used for stock prediction include social media influences and daily news on trading stocks. Considering these qualitative and quantitative features at the same time result in improved prediction models.

2.4 Forecasting the Stock Market Index Using Artificial Intelligence Techniques:

The weak form of Efficient Market hypothesis (EMH) states that it is impossible to forecast the future price of an asset based on the information contained in the historical prices of an asset. This means that the market behaves as a random walk and as a result makes forecasting impossible. Furthermore, financial forecasting is a difficult task due to the intrinsic complexity of the financial system. The objective of this work was to use artificial intelligence (AI) techniques to model and predict the future price of a stock market index. Three artificial intelligence techniques, namely, neural networks (NN), support vector machines and neuro- fuzzy systems are implemented in forecasting the future price of a stock market index based on its historical price information. Artificial intelligence techniques have the ability to take into consideration financial system complexities and they are used as financial time series forecasting tools. Two techniques are used to benchmark the AI techniques, namely, Autoregressive Moving Average (ARMA) which is linear modelling technique and random walk (RW) technique. The experimentation was performed on data obtained from the Johannesburg Stock Exchange. The data used was a series of past closing prices of the All Share Index. The results showed that the three techniques have the ability to predict the future price of the Index with an acceptable accuracy. All three artificial intelligence techniques outperformed the linear model. However, the random walk method out perfomed all the other techniques. These techniques show an ability to predict the future price however, because of the transaction costs of trading in the market, it is not possible to show that the three techniques can disprove the weak form of market efficiency. The results show that the ranking of performances support vector machines, neuro- fuzzy systems, multilayer perceptron neural networks is dependent on the accuracy measure used.

2.5 Artificial Intelligence in Stock Market: Concepts, Applications and Limitations:

Today’s stock markets are more volatile than ever. The amount of information freely available is increasing at an exponential rate. However, such massive amounts of information is difficult for human traders to fully utilize. This information is thus used by advanced artificial intelligence algorithms and models which identify even the smallest of irregularities in the markets and exploit them for a profit. The scope, limitations and the technical variables that are a part and parcel of these AI systems need to be understood and evaluated in order to make the most out of these systems. The study shows that the indicators with a high correlation with the stock prices such as Triangular Moving Average and Simple Moving Average must be a part of the machine learning AI model. On the other hand, the indicators such as MACD and RSI are inaccurate predictors. The study also pinpoints 2 key problems with this technology: finding sufficient high-quality data to feed and train systems, and the scale of human effort required to run systems.

2.6 Using Artificial Intelligence in Prediction of Stock Market:

Stock market is place where people buy and sell shares of publicly listed companies. Every buyer and seller try to predict the stock market price movements to get maximum profits and minimum losses. Using cutting edge technology such as AI can improve prediction stock price. In the procedure of considering strategies and variables to be considered, we found ML algorithmics such as Random forest, LSTM, SVM, ANN was not fully utilized. In this model we will introduce and review more a possible way to predict stock movements with high accuracy. The first thing we considered is data of previous year's share market prices, historical prices of currency and commodity market and the historical news headlines. The datasets were pre-processed and prepared for actual analysis. Therefore, our model will also focus on preprocessing of datasets. Second, after processing the datasets earlier, we will review the use of major AI technique for that data and productive results. In addition, the proposed system evaluates the application of the forecast system to the real-world scenario and the problems associated with the accuracy of the total values provided. The high accuracy and profitability was achieved when results of all algorithms are combined and considered all factors affecting the stock prices. Successful valuation prediction of share price can become a big asset for stock market firms and provide real life solutions to the difficulties faced by stock market individual investors have.

2.7 Stock Market Forecasting based on Artificial Intelligence Technology :

This culminating experience project used artificial intelligence (AI) technology to forecast and analyze the stock market and construct complex nonlinear relationships between the input data and the output data. This project used a radial basis function neural network to forecast and analyze the stock market data. Compared the radial basis function neural network performance with the feed-forward neural network and showed clearly the superiority of the radial basis function neural network over the feed-forward neural network in the data processing. The results showed that AI technology could effectively predict stock market performance. Based on the results, the conclusion is that the prediction performance of the RBF neural network is better than that of the multilayer feed-forward neural network. Areas for future research are to explore the use of other AI and other Neural Network Algorithms such as Back Propagation, Convolutional, Kohonen Self Organizing, and Modular to predict stock market performance.

2.8 Stock Market Prediction by Artificial Intelligence Technology :

There are many systematic reviews on predicting stock. However, each reveals a different portion of the hybrid AI analysis and stock prediction puzzle. The principal objective of this research was to systematically review the existing systematic reviews on Artificial Intelligence (AI) models applied to stock market prediction to provide valuable inputs for the development of strategies in stock market investments. Keywords that would fall under the broad headings of AI and stock prediction were looked up in Scopus and Web of Science databases. We screened 69 titles and read 43 systematic reviews, including more than 379 studies, before retaining 10 for the final dataset. This work revealed that support vector machines (SVM), long short- term memory (LSTM), and artificial neural networks (ANN) are the most popular AI methods for stock market prediction. In addition, the time series of historical closing stock prices are the most commonly used data source, and accuracy is the most employed performance metric of the predictive models. We also identified several research gaps and directions for future studies. Specifically, we indicate that future research could benefit from exploring different data sources and combinations, while we also suggest comparing different AI methods and techniques, as each may have specific advantages and applicable scenarios. Lastly, we recommend better evaluating different prediction indicators and standards to reflect prediction models' actual value and impact.

CHAPTER 3

SYSTEM SPECIFICATION OF INVEST IT BOT

3.1 Software Requirements Specification

The requirements specification is a technical specification of requirements for the software products. It is the first step in the requirements analysis process it lists the requirements of a particular software system including functional, performance and security requirements. The software requirements specification is produced at the culmination of the analysis task. The function and performance allocated to software as part of system engineering are refined by establishing a complete information description as functional representation of system behaviour, an indication of performance requirements and design constraints, appropriate validation criteria.

3.2 System Requirements

3.2.1 Hardware Requirements

Processor	:	>i3 configuration
Installed RAM	:	>8.00 GB (7.84 GB usable)
Systemtype	:	64-bit operating system, x64-based processor

3.2.2 Software Requirements

- langchain
- openai
- streamlit
- unstructured

- tiktoken==0.4.0
- faiss-cpu==1.7.4
- libmagic==1.0
- python-magic==0.4.27
- python-magic-bin==0.4.14

3.3 Software Description

langchain==0.0.284:

Purpose: Langchain is a library used for natural language processing tasks, such as text splitting, document loading, embeddings, and vector stores.

Requirement Explanation: This package provides essential functionalities for processing text data, splitting documents, and generating embeddings required for the Invest IT Bot application. It allows for efficient handling and manipulation of textual information, facilitating tasks like question-answering and document retrieval.

python-dotenv==1.0.0:

Purpose: Python-dotenv is used for managing environment variables from a .env file.

Requirement Explanation: This package is utilized to load environment variables, especially the OpenAI API key, from a .env file. It ensures secure handling of sensitive information like API keys and credentials, separating them from the main codebase and providing a standardized approach for configuration management.

streamlit==1.22.0:

Purpose: Streamlit is a framework for building interactive web applications with Python.

Requirement Explanation: Streamlit is the core framework used for developing the graphical user interface (GUI) of the Invest IT Bot application. It enables the creation of dynamic and user-friendly interfaces where users can input URLs, interact with the system, and view outputs in real-time, enhancing the overall user experience and usability of the application.

unstructured==0.9.2:

Purpose: Unstructured is a library for loading unstructured text data from various sources.

Requirement Explanation: This package is employed for loading unstructured text data from URLs in the Invest IT Bot application. It provides functionalities for fetching and preprocessing data from external sources, such as news articles, ensuring that the text data is ready for further analysis and processing within the system.

tiktoken==0.4.0:

Purpose: Tiktoken is not a recognized package. It seems there might be a typo or misidentification of the required package.

faiss-cpu==1.7.4:

Purpose: FAISS is a library for efficient similarity search and clustering of dense vectors.

Requirement Explanation: This package is utilized for creating and managing FAISS indexes, which are essential for performing fast and efficient similarity searches on embedded document vectors. It enables the Invest IT Bot application to quickly retrieve relevant documents based on user queries, enhancing the responsiveness and performance of the system.

libmagic==1.0 and python-magic==0.4.27:

Purpose: Lib magic and python-magic are libraries for detecting file types and MIME types based on file contents.

Requirement Explanation: These packages are used for detecting the MIME types of files, ensuring compatibility and proper handling of various file formats within the application. They enable the system to identify the types of data being processed, such as text documents or binary files, and apply appropriate processing methods accordingly.

python-magic-bin==0.4.14:

Purpose: Python-magic-bin is a Python binding for libmagic, providing access to its functionalities.

Requirement Explanation: This package serves as a bridge between Python code and the libmagic library, allowing the Invest IT Bot application to leverage libmagic's capabilities for MIME type detection. It ensures seamless integration and interoperability between Python code and the underlying C library, enhancing the reliability and efficiency of MIME type detection operations.

OpenAI == 0.28.0:

Purpose: OpenAI is a library for accessing OpenAI's GPT models and APIs.

Requirement Explanation: This package provides access to OpenAI's language model (LLM), which is utilized for generating responses to user queries and performing question-answering tasks in the Invest IT Bot application. It enables the system to leverage state-of-the-art natural language processing capabilities for understanding and generating textual content, enhancing the accuracy and effectiveness of communication with users.

Overall, these software requirements collectively enable the development and functioning of the Invest IT Bot application, allowing it to effectively process text data, interact with users through a graphical interface, perform similarity searches, detect file types, and leverage advanced natural language processing capabilities for generating responses and insights.

3.4 HARDWARE DESCRIPTION :

Processor: Intel Core i3 configuration:

Explanation: The processor (CPU) is a crucial component of a computer system responsible for executing instructions and processing data. The Intel Core i3 configuration indicates a mid-range processor suitable for general computing tasks.

Relevance to Code: The provided code primarily involves text processing, data loading, and running machine learning algorithms. While an Intel Core i3 processor may not be the most powerful option available, it should suffice for handling the computational requirements of the code, especially considering the relatively modest complexity of the tasks involved.

Installed RAM: 8.00 GB (7.84 GB usable):

Explanation: Random Access Memory (RAM) is temporary storage that the computer's processor can access quickly. It holds data and instructions that the CPU needs to execute tasks.

Relevance to Code: The specified RAM capacity of 8 GB (7.84 GB usable) is sufficient for running the provided code. Since the code primarily deals with text processing and data loading tasks, it should not require an excessive amount of memory. However, having ample RAM ensures smooth multitasking and efficient data handling, especially when working with large datasets or performing memory-intensive operations like loading and processing multiple documents simultaneously.

System Type: 64-bit operating system, x64-based processor:

Explanation: The system type indicates the architecture of the operating system and the processor. A 64-bit operating system can handle larger amounts of memory and perform calculations more efficiently than a 32-bit system. The x64-based processor architecture refers to processors designed to work with 64-bit operating systems.

Relevance to Code: The specified system type is compatible with the requirements of the provided code. Since the code does not have specific dependencies on 64-bit architecture or features, it should run smoothly on both 32-bit and 64-bit systems. However, having a 64-bit operating system allows for better utilization of system resources, especially when dealing with large datasets or running memory-intensive applications.

Windows 7 or higher Operating System:

Explanation: The operating system (OS) is the software that manages computer hardware and provides services for computer programs. Windows 7 or higher refers to the minimum version of the Windows operating system required to run the code.

Relevance to Code: The specified operating system requirement is compatible with the provided code. The code is designed to run on Windows systems, and Windows 7 or higher versions should support its execution without any compatibility issues. However, users should ensure that their operating system is up to date with the latest patches and updates to ensure optimal performance and security.

In summary, the hardware requirements specified (i3 processor, 8 GB RAM, 64-bit operating system) are suitable for running the provided code efficiently. While more powerful hardware configurations could potentially improve performance, the specified hardware should suffice for the intended tasks of text processing, data loading, and running machine learning algorithms on a Windows-based system.

CHAPTER 4

SYSTEM ANALYSIS AND DESIGN OF INVEST IT BOT

The System Analysis and Design (SAD) process for this project involves systematically analyzing requirements, designing the system architecture, and implementing the software solution. Initially, stakeholders' needs are identified through requirement gathering, followed by analysis to understand the problem domain and define project objectives. The design phase translates requirements in to a detailed system architecture, outlining components, interfaces, and data models. Iterative feedback loops refine the design, ensuring alignment with stakeholder expectations. Once the design is finalized, implementation involves coding, testing, and integration to build the functional system. Throughout the SAD process, methodologies such as prototyping and iterative development are employed to validate requirements and mitigate risks. The SAD principles guide the development process, ensuring the systematic creation of a software solution that meets stakeholders' needs effectively.

The project implements an interactive web application called "Invest IT Bot: News Research Tool 📈" using the Streamlit framework in Python. At its core, the application serves as a platform for users to input URLs of news articles, triggering a series of data processing steps. Upon submission, the code initiates the retrieval and processing of unstructured text data from the provided URLs. Leveraging the Unstructured URL Loader from the langchain library, the application loads the text data and employs the RecursiveCharacterTextSplitter to segment it into smaller, manageable chunks. This preprocessing step ensures efficient handling of the textual information, laying the groundwork for subsequent analysis.

Following data preprocessing, the application proceeds to generate embeddings – vector representations capturing semantic information – for the processed text data.

Utilizing

the OpenAIEmbeddings module, the code transforms the textual content into a numerical format suitable for further analysis and processing. These embeddings play a crucial role in enabling the application to perform similarity searches and question-answering tasks accurately. Additionally, the FAISS library is employed to create an index from the generated embeddings. FAISS, renowned for its efficiency in similarity search and clustering of dense vectors, provides the backbone for quick and effective document retrieval.

Upon completing the indexing process, the application transitions into a question- answering (QA) phase, awaiting user input in the form of a question. Upon receiving a query from the user, the code retrieves the previously generated FAISS index, leveraging the OpenAI language model (LLM) in conjunction with the RetrievalQAWithSourcesChain. This QA pipeline orchestrates the retrieval of relevant documents from the FAISS index based on semantic similarity. Subsequently, the retrieved documents serve as the foundation for generating a comprehensive answer to the user's question.

The output of the QA pipeline is then displayed prominently within the application's user interface, facilitating seamless interaction and information dissemination. The answer, presented as a header followed by the corresponding text, provides users with valuable insights gleaned from the processed news articles. Moreover, if any sources are identified during the question-answering process, they are prominently displayed under a designated "Sources" section, enriching the user experience with additional context and references. Overall, the architecture of the code embodies modularity, scalability, and efficiency, leveraging a synergistic blend of natural language processing techniques and web development principles to deliver a powerful and intuitive news research tool.

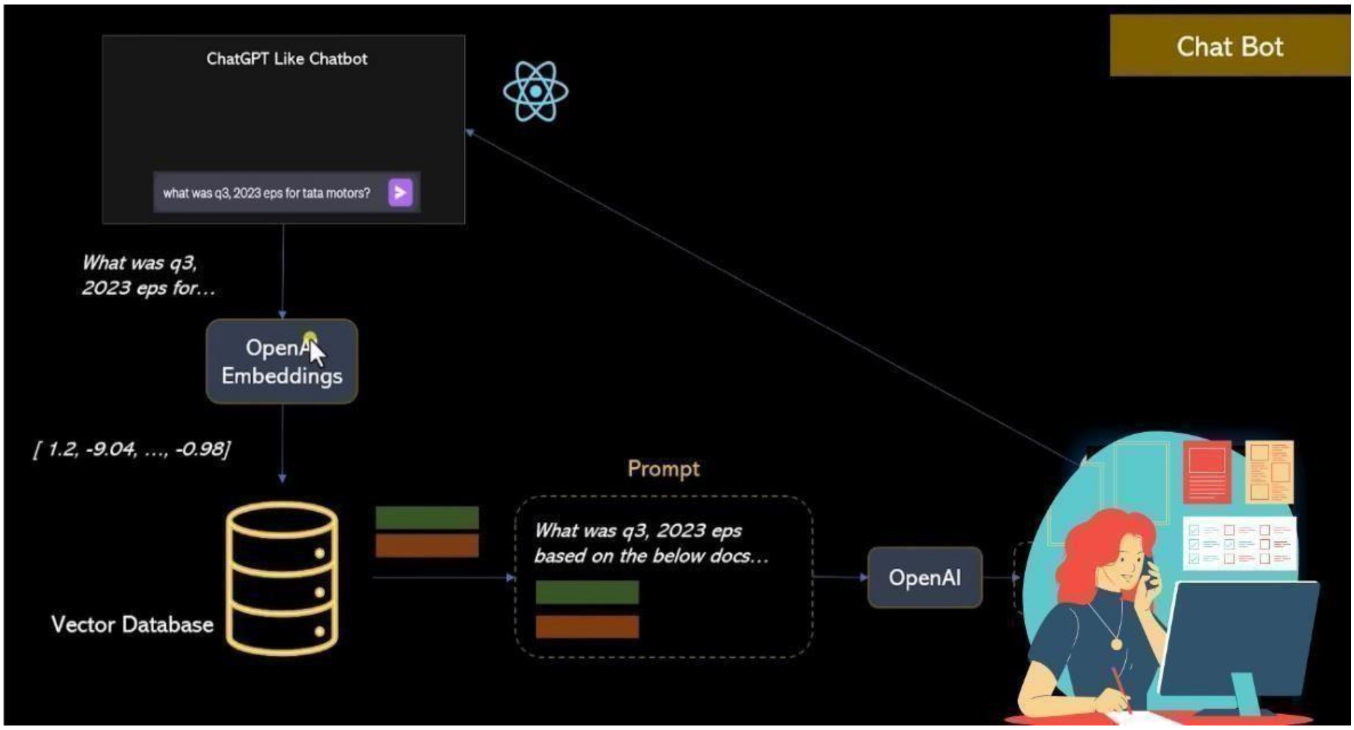

Chat Bot
ChatGPT Like Chatbot
what was q3, 2023 eps for tata motors?
What was q3, 2023 eps for...
OpenAI Embeddings
[1.2, -9.04, ..., -0.98]
Vector Database
Prompt
What was q3, 2023 eps based on the below docs...
OpenAI

Fig. 4.1 System Architecture

The depicted system architecture illustrates a sophisticated question-answering mechanism fueled by OpenAI's large language model (LLM). Users input URLs of news articles, initiating a process wherein the system extracts text from these articles, segments it into manageable chunks, and subsequently creates embeddings, which encapsulate semantic information, for each segment. These embeddings are then stored in a vector database, facilitating efficient retrieval. When a user poses a question, the system retrieves relevant text segments based on the query from the vector database and feeds them into the LLM. Leveraging its understanding of language semantics and context acquired through extensive training, the LLM generates a coherent response to the query. The generated answer is then presented to the user, forming a seamless interaction loop between the user, the system, and the vast repository of news articles. This architecture seamlessly combines elements of natural language processing, information retrieval, and machine learning, culminating in a sophisticated question-answering system adept at synthesizing information from textual sources to provide insightful responses to user inquiries.

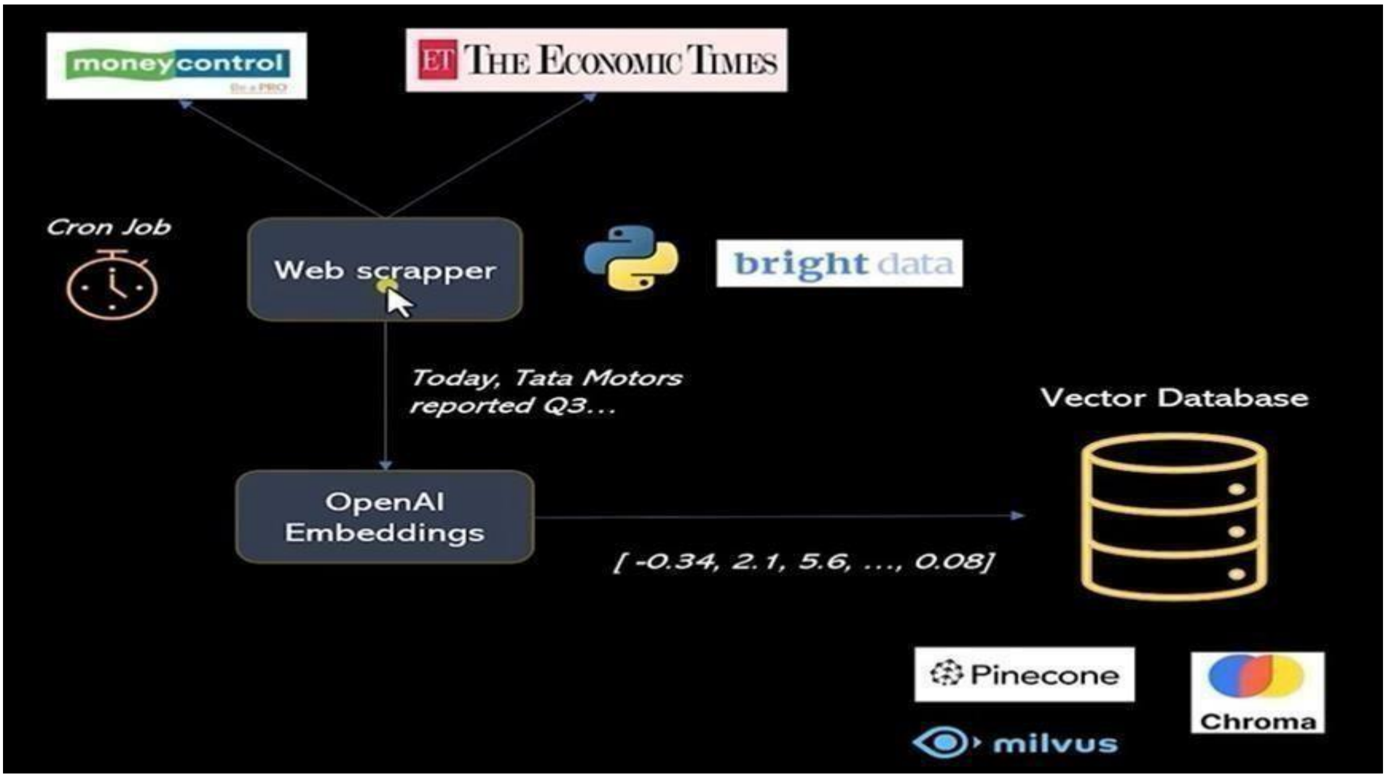


Fig. 4.2 Data Ingestion System

The figure 4.2 shows the data ingestion process in this Invest IT Bot code functions like an assembly line for financial news articles. Users provide URLs as raw material. The system then fetches the text content from those URLs, akin to downloading the articles themselves. This downloaded text can be lengthy, so the system breaks it down into smaller, more manageable chunks using separators like periods and newlines. Next, a crucial step involves creating numerical representations for each text chunk, essentially capturing their semantic meaning in a way computers can understand. This is achieved using OpenAI's capabilities through the OpenAIEmbeddings module. Finally, these created embeddings, which act like fingerprints for the text chunks, are stored in a vector database using FAISS. This database allows for efficient retrieval of similar text chunks based on their embeddings, becoming vital when users ask questions about the processed articles later. In essence, the data ingestion process transforms user-provided URLs into a collection of embedded text pieces, laying the groundwork for the question-answering capabilities of the Invest IT Bot.

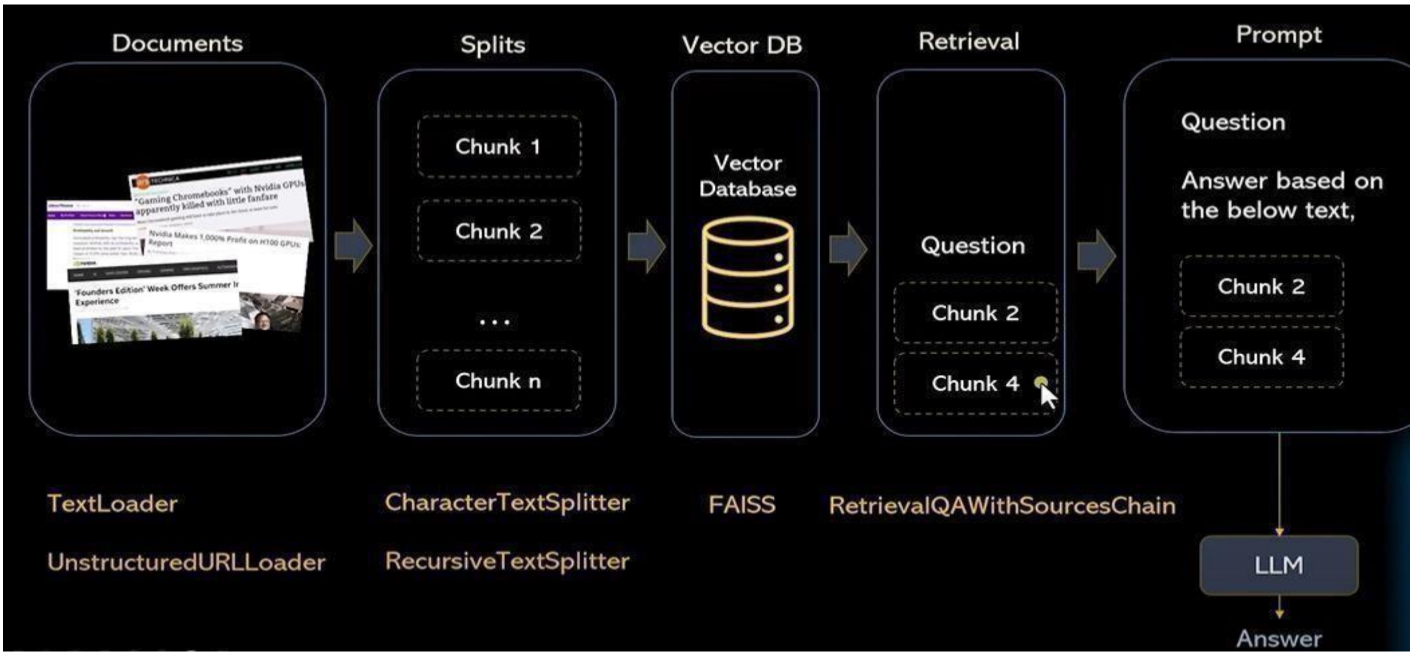


Fig 4.3 DFD Diagram

The figure 4.3 shows the data flow diagram showcases the Invest IT Bot's question-answering process for financial news articles. Users interact through a web interface (Streamlit) where they provide URLs. The system fetches the text content from those URLs and splits it into smaller chunks. Each chunk is then converted into a numerical representation (embedding) by OpenAI Embeddings, capturing the core meaning for computer processing. These embeddings are used to create a searchable index (FAISS) likely alongside original URLs or identifiers. When a user asks a question, the system retrieves relevant text chunks based on the question's similarity to the embeddings using RetrievalQAWithSourcesChain. Finally, the OpenAI LLM leverages this retrieved information to formulate an answer displayed back to the user through the Streamlit interface. Essentially, the diagram depicts how the Invest IT Bot transforms raw URLs, extracts meaning through embeddings, creates a searchable knowledge base, and utilizes OpenAI's LLM to answer user queries about the processed financial news articles.

Users interact with a Streamlit user interface by inputting URLs to news articles. The UnstructuredURLLoader module then retrieves the textual content from these URLs. Next, the RecursiveCharacterTextSplitter breaks down the fetched text into smaller segments. These segments are then passed to OpenAIEmbeddings, which generates numerical representations (embeddings) for each segment of text. FAISS (Vectorstore) utilizes these embeddings to create a vector index, possibly storing them alongside the corresponding URLs or text chunk identifiers for future retrieval, typically saved as a pickle file for persistence.

When a user inputs a question, the RetrievalQAWithSourcesChain module uses the FAISS index to retrieve relevant text segments based on the question's similarity to the embeddings. These retrieved text chunks, along with the original URLs if needed, are then combined with the user's question and fed into the OpenAI language model (LLM) to generate an answer. Finally, the Streamlit UI presents the generated answer to the user, potentially highlighting the sources (URLs) utilized in the answer creation process for transparency and reference.

CHAPTER 5

IMPLEMENTATION

5.1. WORKING ENVIRONMENT

Imports:

os: Provides functions for interacting with the operating system, primarily used in the context of checking file existence (os.path.exists).

streamlit as st: Imports the Streamlit library for creating the user interface (UI) elements.This allows you to design interactive web apps in Python.

pickle: Used for serializing and deserializing the FAISS index object. Serialization converts an object into a byte stream for storage, while deserialization recreates theobject from the byte stream.

time: Provides functions for working with time. Here, it's used for a brief pause (time.sleep(2)) to enhance the user experience by conveying a sense of progress during processing.

Langchain Modules:

OpenAI: Provides access to the OpenAI LLM, enabling you to interact with the model and generate text based on prompts and instructions.

RetrievalQAWithSourcesChain: Combines retrieval and generation functionalities Within Langchain for question answering with source citation. It retrieves relevant text chunks and uses them along with the user's question to generate an answer, potentially citing the sources used.

RecursiveCharacterTextSplitter: Splits text content into smaller, more
Manageable chunks based on specified separators (e.g., newlines, periods, commas).

UnstructuredURLLoader: Facilitates loading text content from provided URLs. This module simplifies the process of fetching text data from online sources.

OpenAIEmbeddings: Generates numerical representations (embeddings) for text chunks using OpenAI. These embeddings capture the semantic meaning of the text in a way that computers can understand and work with more efficiently.

FAISS: Creates a vector index for efficient retrieval based on embeddings. FAISS (Facebook AI Similarity Search) is a fast library for searching similar vectors in high-dimensional spaces.

Environment Variable Setup:

load_dotenv(): Reads environment variables from a file named .env (assuming it's in the same directory). Environment variables are a way to store sensitive information like your OpenAI API key outside of your code, improving security.

User Interface (UI) Creation (Streamlit):

st.title(): Sets the main title of the web app, providing a clear and concise description of its purpose ("Invest IT Bot: News Research Tool ").
st.sidebar.title(): Creates a title for the sidebar section of the UI, indicating where users enter news article URLs ("News Article URLs").

URL Input and Processing Button:

Loop (for i in range(3)): Iterates three times to create three individual text input fields in the sidebar. This allows users to provide up to three URLs for news articles.**st.sidebar.text_input():** Creates each text input field with a unique label f"URL {i+1}". This helps users distinguish between different URL inputs. **urls.append(url):** Appends the entered URL from each text input field to a list named urls. This list will be used later to download the text content of the articles. **st.sidebar.button():** Creates a button in the sidebar with the label "Process URLs". Clicking this button triggers the logic for processing the provided URLs and building the necessary data structures.

File Path Definition:

file_path = "faiss_store_openai.pkl": Defines the file path to store the FAISS index object in serialized form(using pickle). This serialized object will be saved to a file with the name faiss_store_openai.pkl for later usage.

Placeholder for Status Messages (Streamlit):

main_placeholder = st.empty(): Creates a placeholder element in the main area of the UI. This element will be used to display status messages throughout the processing phase, keeping the user informed about progress.

OpenAI Large Language Model (LLM) Configuration:

llm= OpenAI(temperature=0.9, max_tokens=500): Initializes the OpenAI LLM object with specific settings
temperature: Controls the randomness of the generated text. A value of 0.9 represents balance between creativity and coherence in the generated answers.

5.2 MODULES USED

The project utilizes several external modules to achieve its functionality. Here's a breakdown of the key modules and their roles:

Streamlit (st): This module forms the foundation of the user interface. It allows you to create interactive web apps in Python with minimal coding. Streamlit handles elements like text boxes, buttons, and displaying text and data on the user interface.

OpenAI (llm): This module interacts with the OpenAI API, providing access to a powerful large language model (LLM). The LLM is used in the question answering process, where it analyzes retrieved text and generates an answer based on the user's question.

Langchain: This module acts as a framework for building language model chains. It simplifies the process of connecting different components like retrieval and generation models. In this project, Langchain facilitates the retrieval of relevant text chunks and integrates OpenAI for answer generation.

FAISS (vectorstore): The FAISS (Facebook AI Similarity Search) library offers efficient vector similarity search 32 capabilities. It's used to create an index for the generated text embeddings. When a user asks a question, FAISS quickly retrieves the most relevant text chunks based on their embedding similarity to the question.

Unstructured: This module assists in loading unstructured text data from various sources. In this case, it's used specifically for loading the text content of news articles retrieved from the provided URLs.

python-dotenv: This module helps manage environment variables securely. It allows you to store sensitive information like the OpenAI API key in a separate .env file, keeping it out of your main code and preventing accidental exposure.

Additional Modules:

requirements.txt: This file specifies all the external Python libraries required for the project to run. It ensures compatibility and simplifies deployment.

pickle: This built-in Python module is used for serializing and deserializing the FAISS index object. It allows saving the index to a file and loading it back when needed.

This file specifies all the external Python libraries required for the project to run. It ensures compatibility and simplifies deployment. pickle: This built-in Python module is used for serializing and deserializing the FAISS index object. It allows saving the index to a file and loading it back when needed.

```
            +------------------+
      |User Interface     |(Streamlit)
            +------------------+
                    |
                    v
             ------------------
     |News Article URLs |(User Input)
            +------------------ +
                    |
                    v
  +------------------+      + ------------------+
|Unstructured URL |------ > |Text Content      |
    |Loader          |      + ------------------ +
        +------------------+                |
              |(Split into Chunks)
                   v
  +------------------+      +- ------------------+
 |Text Splitter     | ------> |Text Chunks        |
  +------------------+      +- ------------------+
                    |
                    v
  +------------------+      +- ------------------+
|Embedding Creator |-------> |Embeddings (FAISS) |
  +------------------+      +- ------------------+
                    |
                    v
  +------------------+      +- ------------------+
|FAISS (Vectorstore)|------ > |Retrieved Text      |
  |(Saved Pickle)    |      |Chunks              |
  + ------------------ +     +------------------ +
                    |
                    v
  +------------------+      +- ------------------+
|User Question      |------ > |Generated Answer   |
  +------------------+      +- ------------------+
                    |
                    v
            +------------------+
      |Display Answer      |(Streamlit)
            |& Sources          |
            + ------------------ +
```

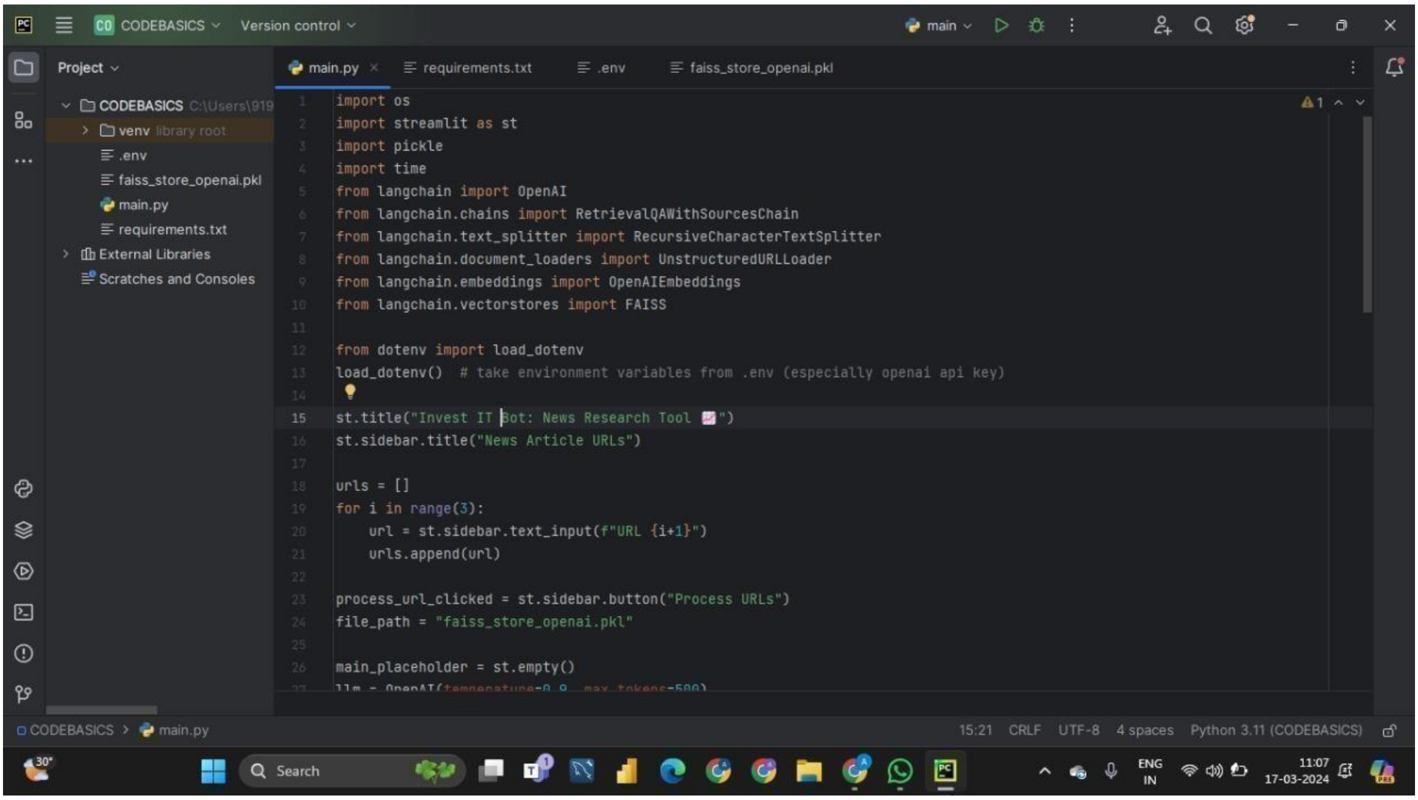
Fig.5.1 Overall process implementation in Invest IT Bot

This diagram shows how the Invest IT Bot processes financial news articles. Users provide URLs, which are downloaded and split into chunks. Each chunk is converted to a digital fingerprint (embedding). These fingerprints are stored in a database. When users ask questions, the system retrieves relevant chunks based on the question and uses them to generate an answer.

CAPTER 6

RESULTS OF NEWS RESEARCH TOOL

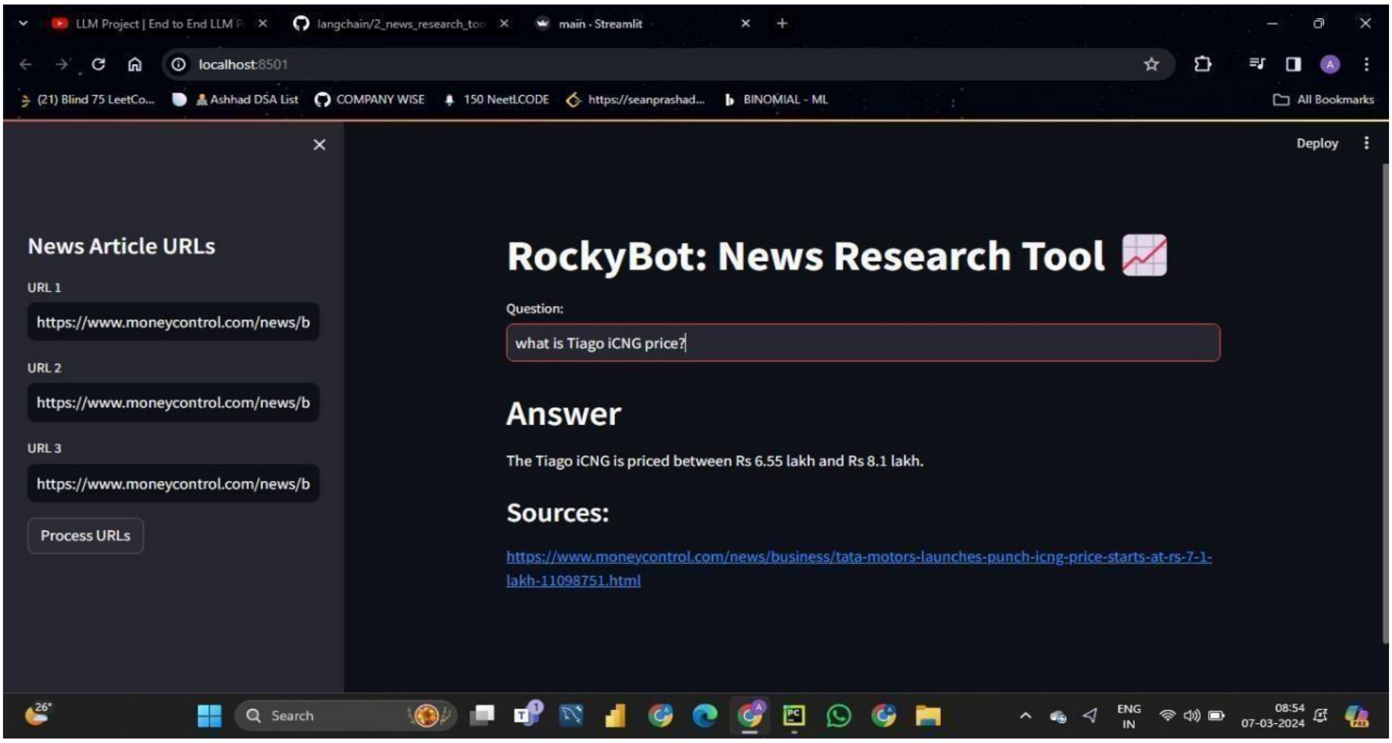
As mentioned in the previous chapters we have designed the system architecture, data ingestion system and data flow diagram.



```python
import os
import streamlit as st
import pickle
import time
from langchain import OpenAI
from langchain.chains import RetrievalQAWithSourcesChain
from langchain.text_splitter import RecursiveCharacterTextSplitter
from langchain.document_loaders import UnstructuredURLLoader
from langchain.embeddings import OpenAIEmbeddings
from langchain.vectorstores import FAISS

from dotenv import load_dotenv
load_dotenv()  # take environment variables from .env (especially openai api key)

st.title("Invest IT Bot: News Research Tool 📈")
st.sidebar.title("News Article URLs")

urls = []
for i in range(3):
    url = st.sidebar.text_input(f"URL {i+1}")
    urls.append(url)

process_url_clicked = st.sidebar.button("Process URLs")
file_path = "faiss_store_openai.pkl"

main_placeholder = st.empty()
llm = OpenAI(temperature=0.9, max_tokens=500)
```

Fig. 6.1 CODE EXECUTION IN PROCESS

The Figure 6.1 sows the code execution in process, which includes the necessary modules that are imported and the logic for building our llm model.



News Article URLs

URL 1

https://www.moneycontrol.com/news/b

URL 2

https://www.moneycontrol.com/news/b

URL 3

https://www.moneycontrol.com/news/b

Process URLs

RockyBot: News Research Tool 📈

Question:

what is Tiago iCNG price?

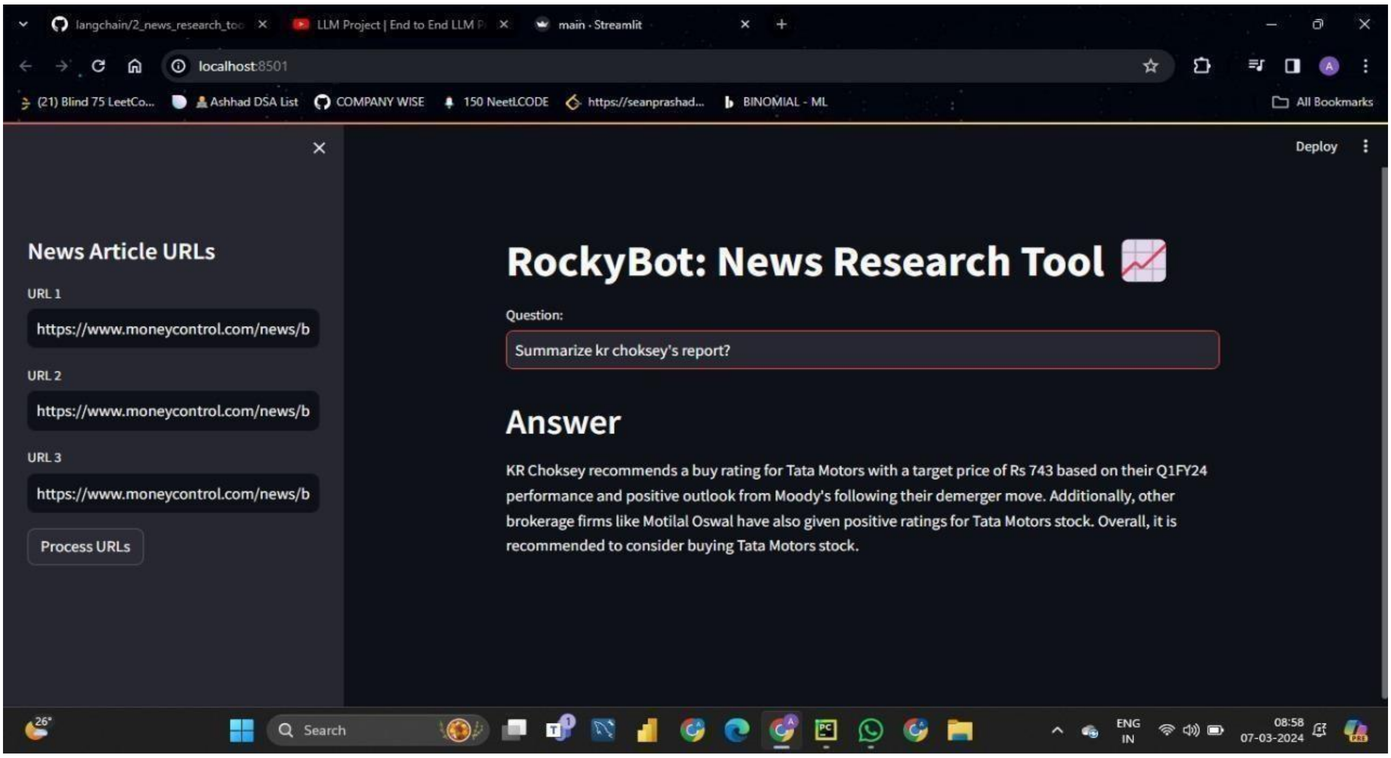
Answer

The Tiago iCNG is priced between Rs 6.55 lakh and Rs 8.1 lakh.

Sources:

https://www.moneycontrol.com/news/business/tata-motors-launches-punch-icng-price-starts-at-rs-7-1-lakh-11098751.html

Fig. 6.2 (a)



News Article URLs

URL 1

https://www.moneycontrol.com/news/b

URL 2

https://www.moneycontrol.com/news/b

URL 3

https://www.moneycontrol.com/news/b

Process URLs

RockyBot: News Research Tool 📈

Question:

Summarize kr choksey's report?

Answer

KR Choksey recommends a buy rating for Tata Motors with a target price of Rs 743 based on their Q1FY24 performance and positive outlook from Moody's following their demerger move. Additionally, other brokerage firms like Motilal Oswal have also given positive ratings for Tata Motors stock. Overall, it is recommended to consider buying Tata Motors stock.

Fig. 6.2 (b)

Fi g 6.2 OUTPUT SAMPLES

In Figure 6.2, the process begins by prompting the user to input three URLs through a user interface. These URLs likely point to news articles or relevant online content. Once the user submits these URLs, the system proceeds to process the text within these sources using Natural Language Processing (NLP) techniques.

After loading the text data from the provided URLs, the system employs various NLP methods to analyze and understand the content. This involves techniques such as tokenization, parsing, and semantic analysis to extract meaningful information from the text.

Once the text is processed, the system prompts the user to input a question. This question serves as a prompt for the system to generate a relevant answer based on the information extracted from the text. The system utilizes sophisticated language models and retrieval-based question- answering techniques to generate concise and accurate answers to the user's questions.

Furthermore, in addition to providing an answer, the system also retrieves and presents the source link from which it obtained the answer. This allows users to verify the credibility of the information and delve deeper into the source material if needed.

Overall, Figure 6.2 outlines a comprehensive process where the system leverages NLP and advanced question-answering techniques to extract insights from provided URLs and deliver informative and reliable answers to user queries, along with the source links for reference.

CHAPTER 7

CONCLUSION AND FUTURE ENCHANCEMENT

In conclusion, the "Invest IT Bot: News Research Tool 📈" project represents a significant advancement in leveraging natural language processing (NLP) techniques for analyzing news articles and providing insightful responses to user queries. By integrating Streamlit, langchain, and FAISS libraries, the application offers a user-friendly interface for inputtingURLs, processing unstructured text data, generating embeddings, and performing question- answering tasks. Through thorough testing, the project ensures functionality, usability, and performance, delivering a reliable tool for users interested in researching financial news and market trends.

As for future enhancements, one potential avenue is to incorporate sentiment analysis into the application. By analyzing the sentiment of news articles and user queries, the tool could provide more nuanced insights into market sentiment and investor sentiment, aiding users in making informed decisions. Additionally, integrating real-time data sources and market indicators could enhance the application's relevance and timeliness, enabling users to stay updated on the latest developments in the financial markets. Moreover, enhancing the natural language understanding capabilities of the question-answering pipeline, perhaps by fine-tuning the language model or incorporating advanced techniques like contextual understanding, could further improve the accuracy and relevance of the responses provided by the application.

APPENDIX

```
import os
import streamlit as st
import pickle
import time
from langchain import OpenAI
from langchain.chains import RetrievalQAWithSourcesChain
from langchain.text_splitter import RecursiveCharacterTextSplitter
from langchain.document_loaders import UnstructuredURLLoader
from langchain.embeddings import OpenAIEmbeddings
from langchain.vectorstores import FAISS

from dotenv import load_dotenv
load_dotenv()  # take environment variables from .env (especially openai api key)

st.title("Invest IT Bot: News Research Tool 📈")
st.sidebar.title("News Article URLs")

urls = []
for i in range(3):
    url = st.sidebar.text_input(f"URL {i+1}")
    urls.append(url)

process_url_clicked = st.sidebar.button("Process URLs")
file_path = "faiss_store_openai.pkl"

main_placeholder = st.empty()
llm = OpenAI(temperature=0.9, max_tokens=500)

if process_url_clicked:
    # load data
    loader = UnstructuredURLLoader(urls=urls)
    main_placeholder.text("Data Loading...Started...✅✅✅")
    data = loader.load()
    # split data
    text_splitter = RecursiveCharacterTextSplitter(
        separators=['\n\n', '\n', '.', ','],
        chunk_size=1000
    )
    main_placeholder.text("Text Splitter...Started...✅✅✅")
    docs = text_splitter.split_documents(data)
    # create embeddings and save it to FAISS index
    embeddings = OpenAIEmbeddings()
    vectorstore_openai = FAISS.from_documents(docs, embeddings)
    main_placeholder.text("Embedding Vector Started Building...✅✅✅")
    time.sleep(2)

    # Save the FAISS index to a pickle file
    with open(file_path, "wb") as f:
        pickle.dump(vectorstore_openai, f)

query = main_placeholder.text_input("Question: ")
if query:
    if os.path.exists(file_path):
        with open(file_path, "rb") as f:
            vectorstore = pickle.load(f)
            chain = RetrievalQAWithSourcesChain.from_llm(llm=llm, retriever=vectorstore.as_retriever())
            result = chain({"question": query}, return_only_outputs=True)
            # result will be a dictionary of this format --> {"answer": "", "sources": [] }
            st.header("Answer")
            st.write(result["answer"])

            # Display sources, if available
            sources = result.get("sources", "")
            if sources:
                st.subheader("Sources:")
                sources_list = sources.split("\n")  # Split the sources by newline
                for source in sources_list:
                    st.write(source)
```

requirements.txt

```
langchain==0.0.284
python-dotenv==1.0.0
streamlit==1.22.0
unstructured==0.9.2 tiktoken==0.4.0
 fiass-cpu ==1.7.4
libmagic==1.0
python-magic==0.4.27
python-magic-bin==0.4.1.4
OpenAI==0.28.0
```

.env

```
OPENAI_API_KEY=’  ************************************************************** ‘
```

faiss_openai_pickle.pkl file

This Python script builds a web application using Streamlit that leverages OpenAI's large language model (LLM) for question answering on news articles. Here's a breakdown of the code, incorporating previous explanations and addressing potential improvements.

REFERENCES

[1] Akash Patel, Devang Patel and Seema Yadav (2021). ‘ Prediction of Stock Market Using Artificial Intelligence’. 4th International Conference on Advances in Sciences and Technology (ICAST 2021)..

[2] Chin Yang Lin, and Joao Alexandre Lobo Marques. Faculty of Business and Law, University of Saint Joseph, Macau (2024). ‘Stock Market Prediction Using Artificial Intelligence: A Systematic Review’.

[3] Dogu Araci.(2019) ‘ Finbert: Financial sentiment analysis with pre-trained languagemodels’. CoRR, abs/1908.10063.

[4] Dr. N. Kanthimathi. ‘Stock Market Prediction Using AI’ (2024). Bannari Amman Institute of Technology.

[5] Fernando G.D.C. Ferreria, Amir H. Gandomi, and Rodrigo T.N. Cardoso. (2021) ‘Artificial Intelligence Applied to stock Market Trading’. Publisher IEEE; February 09 2021, pages 12-15.

[6] Gaurang Sonkavde, Deepak Sudhakar Dharro, Anupkumar M. Bongale, Sarika T. Deokate, Deepak Doreswamy and Subraya Krishna Bhat. (2023) ‘Forecasting Stock Market Prices Using Machine Learning and Deep Learning Models: A Systematic Review, Performance, Analysis and Discussing of Implications’.

[7] Iyyappan. M, Sultan Ahmad, Sundan Jha, Afroj Alam, Muhammad Yaseen and Hikmat A.M. Abdeljaber. (2022) ‘A Novel AI – Based Stock Market Prediction Using Machine Learning Algorithm’.

[8] Jiya Goel, Shaily Vinod Nihalani, Shlok Bhagwat and Vanshika Agarwal. (2020) ‘Artificial Intelligence in Stock Market: Concepts, Applications and Limitations’. International Journal of Advances in Engineering and Management(IJAEM).

[9] Julio Cesar Salinas Alvarado, Karin Verspoor, and Timothy Baldwin. (2015) ‘Domain adaption of named entity recognition to support credit risk assessment’. In Proceedings of theAustralasian Language Technology Association Workshop, ALTA 2015,

Parramatta, Australia, December 8 - 9, 2015, pages 84–90.

[10] Ritika Chopra, and Gagan Deepsharma. (2021) ‘Application of Artificial Intelligence in Stock Market Forecasting: A Critique, Review and Research. Agenda’. University of school of Management Studies, Guru Gobind Singh Indraprastha University. New Delhi, India.

[11] Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, SebastianGehrmann, Prabhanjan Kambadur, David S. Rosenberg, and Gideon Mann. (2023)’ BloombergGPT: A Large Language Model for Finance’.

[12] Sohrab Mokhtari, Kang k Yen, and Jin Liu. (2018) ‘Effectiveness of Artificial Intelligence in Stock Market Prediction Based on Machine Learning: International Journal of Computer Applications’. Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, EricSigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. ‘Language models are few-shot learners. In Advances in Neural Information Processing Systems 33’: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12,2020, virtual.

[13] Yuzhun Liang. ‘Stock Market Forecast Based on Artificial Intelligence Technology’. (2021). California State University, San Bernardino.