AgroHaven

Project report submitted to

Indian Institute of Information Technology, Nagpur,

in partial fulfillment of the requirements for the Mini Project - I

at

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

By

Ayush Parwal

Hriday Sekhani

Fazl Omar

Shreyash Verma

Under the Supervision of

Mr. Adil Khan

Session Period: July to Dec 2024

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, NAGPUR

(An Institution of National Importance by Act of Parliament) Survey No.140,141/1, Waranga, Nagpur, 441108

Declaration

I/We, **Ayush Parwal, Hriday Sekhani, Fazl Omar, Shreyash Verma**, hereby declare that this project work titled "**AgroHaven : Urban Farming**" is carried out by me/us in the Department of Computer Science and Engineering of Indian Institute of Information Technology, Nagpur. The work is original and has not been submitted earlier whole or in part for the award of any other certification programme at this or any other Institution /University.

Date:	**Sr. No.**	**Names**	**Signature**

Certificate

This is to certify that the project titled "**AgroHaven : Urban Farming**", submitted by **Ayush Parwal, Hriday Sekhani, Fazl Omar, Shreyash Verma** in partial fulfillment of the requirements for the Mini-Project in CSE department IIIT Nagpur. The work is comprehensive, complete and fit for final evaluation.

Date:

Name of the Supervisor
Designation, Dept, IIIT, Nagpur

Acknowledgement:

The journey of completing this mini-project has been both challenging and enlightening, and it would not have been possible without the support, guidance, and contributions of several individuals and organizations.

First and foremost, we would like to express our deepest gratitude to Mr. Adil Khan, our project supervisor from the Department of Computer Science and Engineering at the Indian Institute of Information Technology, Nagpur. His invaluable guidance, constructive feedback, and unwavering support throughout the project have been instrumental in shaping this research work.

We are profoundly thankful to the Indian Institute of Information Technology, Nagpur, for providing the infrastructure, resources, and academic environment that enabled us to pursue this innovative project. The institute's commitment to technological advancement and research has been a constant source of inspiration.

Special acknowledgment is due to the open-source community and the developers of the critical technologies that formed the backbone of our project. We extend our sincere appreciation to:

Our heartfelt thanks to the researchers and organizations who have developed and shared the pre-trained models and datasets that were crucial to our project's success. The collaborative spirit of the technological and academic communities has been truly remarkable.

To our fellow students and colleagues who provided support, critical feedback, and encouragement during various stages of this project – we are grateful for there insights and camaraderie.

Lastly, we would like to thank our families for their patience, understanding, and consistent support throughout this academic endeavour.

This project stands as a testament to the power of collaborative learning and technological innovation.

AgroHaven

Department of Computer Science and Engineering (AI-ML)

Indian Institute of Information Technology, Nagpur

December 2024

Abstract:

Urban farming is an innovative and sustainable solution to the challenges posed by urbanization, promoting self-reliance, environmental stewardship, and healthier lifestyles. Our project focuses on developing a comprehensive end-to-end website aimed at empowering individuals to start and excel in urban farming while maximizing their profitability. The platform offers detailed, location-specific information to help urban farmers grow crops under optimal conditions tailored to their city or region.

Key factors such as lighting, temperature, available space, and crop types are thoroughly analyzed to provide actionable insights that support informed decision-making.

Additionally, the website is an ecommerce place for people to buy crops and other equipment. Chatbot on the website provide modern techniques to enhance farming productivity and efficiency. It addresses the economic viability of urban farming by advising farmers on the best conditions and practices to cultivate crops for maximum yield and profit.

By offering guidance on transforming urban spaces into thriving, productive green areas, the platform helps urban farmers tap into unused resources while contributing to environmental sustainability. This initiative not only supports economic growth but also encourages urban communities to adopt greener, healthier, and more sustainable living practices.

Table of Contents

List of Figures:-

List of Tables

List of Symbols, Abbreviations or Nomenclature

LSTM :- Long Short Term Memory

RAG: - Retrieval Augmented Generation

ML: - Machine Learning

For Rouge Score:

β: Weight balancing precision and recall.

For Bleu Score:

Where:

- P_n: Precision for n-grams (ratio of overlapping n-grams between candidate and reference).
- W_n: Weight for each n-gram.
- BP: Brevity Penalty to penalize overly short translations.

Brevity Penalty (BP):

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ \exp(1 - \frac{r}{c}) & \text{if } c \leq r \end{cases}$$

- c: Length of candidate text.
- r: Length of reference text.

1)Introduction

Urban farming represents a transformative approach to food production within urban environments, addressing the challenges posed by rapid urbanization. This innovative practice involves cultivating, processing, and distributing food in cities, utilizing underused spaces like rooftops and vacant lots.

By integrating sustainable agricultural methods, urban farming promotes self-reliance and environmental stewardship while enhancing community health. Our project aims to develop a comprehensive website that empowers individuals to engage in urban farming effectively.

The platform will provide detailed, location-specific information tailored to various cities, ensuring that urban farmers can optimize their growing conditions based on factors such as lighting, temperature, and available space. To further enhance productivity, a chatbot will offer modern farming techniques and strategies to maximize yield and profitability. By guiding users on how to transform urban spaces into productive green areas, this initiative not only supports economic growth but also encourages sustainable living practices within urban communities. Ultimately, this project aspires to create thriving urban farms that contribute to food security and environmental sustainability while fostering healthier lifestyles.

1.1 Background of Urban Farming

Urban farming, also known as urban agriculture, involves the cultivation of food within urban spaces to support local communities, enhance food security, and promote environmental sustainability. It utilizes unconventional spaces like rooftops, vacant lots, and even vertical walls to produce food, catering to the growing population in cities. The practice has gained momentum due to rising concerns about food supply chains, environmental degradation, and the need for resilient agricultural systems.

Key Benefits of Urban Farming

1. Localized Food Production: Reduces the carbon footprint by minimizing transportation.
2. Efficient Use of Resources: Incorporates sustainable methods like hydroponics, aquaponics, and vertical farming.
3. Environmental Advantages: Mitigates urban heat effects, improves air quality, and promotes biodiversity.
4. Community and Economic Impact: Provides opportunities for education, job creation, and community engagement.

Challenges in Urban Farming

1. Space Scarcity: Urban areas often lack sufficient arable land for large-scale farming.
2. Soil and Water Quality: Urban soils may contain pollutants, and water availability can be inconsistent.
3. Energy Costs: Advanced techniques like hydroponics require significant energy input.
4. Regulations: Zoning laws and land-use restrictions may limit urban farming activities.

Machine Learning Models in Urban Farming

Machine Learning (ML) has been increasingly applied to optimize urban farming practices, addressing challenges like resource management, crop yield prediction, and pest control. Below are some notable applications and ML models implemented in urban farming:

1. Crop Monitoring and Health Assessment

1. Identifying plant diseases using image classification.
2. Monitoring crop health using drone or IoT camera feeds.
3. Example: Using VGG16 to classify images of diseased plants in urban farms.

2. Yield Prediction

- Models Used: Random Forest, Gradient Boosting Machines (e.g., XGBoost, LightGBM), Long Short-Term Memory (LSTM) Networks
- Applications:
    - Predicting crop yield based on environmental factors such as temperature, humidity, and light intensity.
    - Optimizing the allocation of resources like water and nutrients for maximum yield.

3. Resource Optimization

- Models Used: Reinforcement Learning (RL), Linear Regression, and Decision Trees
- Applications:
    - Optimizing water usage through predictive irrigation systems.
    - Managing energy consumption in controlled environments like vertical farms.

4. Pest and Disease Management

- Models Used: Support Vector Machines (SVM), YOLO (You Only Look Once), and Faster R-CNN
- Applications:
    - Detecting pests in real-time using image and video feeds.
    - Predicting disease outbreaks using climate and soil data.

5. Climate Control in Vertical Farms

- Models Used: K-Means Clustering, LSTM, and ARIMA for Time Series Analysis
- Applications:
    - Controlling temperature, humidity, and light in controlled environments to optimize plant growth.
    - Predicting climate fluctuations and their impact on urban farms.

6. Supply Chain and Market Analysis

- Models Used: Recurrent Neural Networks (RNNs), Natural Language Processing (NLP) Models
- Applications:
    - Forecasting demand for specific crops in urban markets.
    - Optimizing supply chain logistics for urban farm produce.

1.2 Problem Statement

Specific Challenges in Urban Farming

1)Urban farming is a relatively new concept, and there has been limited research or work done in this area. As a result, individuals who wish to pursue urban farming, either as a career or a hobby, often lack adequate resources and guidance.

2)Enthusiasts are often uncertain about key factors such as the amount of space or land required, the optimal temperature conditions, and the resources necessary to implement urban farming effectively. This lack of clarity discourages them from taking the first step.

3) Currently, there are no well-established communities or networks that actively promote urban farming or contribute to its growth as a sustainable concept. This absence of support hampers collaboration and sharing of knowledge among interested individuals.

4)There is a need for an end-to-end platform that fosters collaboration among urban farming enthusiasts, providing them with easy access to all necessary resources. Such a platform could also serve as an e-commerce hub where people can buy and sell crops, tools, and farming equipment.

We as team "AgroHaven" are exactly doing this to fill in these gaps to foster urban farming.

1.3 Project Objectives

The primary aim of this project is to design and develop a system capable of leveraging advanced technologies in natural language processing, and linguistic models, the project seeks to address the challenges associated with Urban Farming.

Primary Goals of the Project:

1. **Help Promote Urban Farming.**
2. **Enhance existing methods of Urban Farming:**
    - Bridge the gap between traditional agricultural practices and modern urban farming techniques by integrating innovative methods such as vertical farming with advanced technology and data-driven approaches.
    - Support researchers, educators, and enthusiasts in understanding ancient languages and cultural contexts.
3. **Resource Optimization:**
    - Develop an automated system that minimizes manual effort and human error in extracting and interpreting farming text.
    - Enable scalability for large collections of farming documents.

Specific Technical Objectives

1. **Accurate Price Prediction:**
   Implement a model to accurately predict the prices of crops in the marketplace through the given environment conditions in which it was grown.

2. **Community Development:**
   Develop a community fostering environment on our website to help individuals come up for urban farming and network with like-minded individuals.

3. **Custom Chatbot Development:**
   Train models on Urban Farming documents like books, literature written by scientists, etc. to create a chatbot.

4. **User-Friendly Interface**
   Provide an intuitive platform for urban farmers and consumers to connect and benefit each other with services and money-in-exchange.

5. **Performance Optimization**
   Ensure efficient processing for large prompts, minimizing processing time.

Expected Outcomes

1. **Functional System for Urban Farmers:**
    - A fully operational pipeline that can provide urban farmers with the know-how. Also, how to convert the marketplace to a maximum yield production thereby maximizing their profits.

2. **High Accuracy and Reliability:**
    - Achieve significant accuracy in question-answering and context handling, with minimal errors even for degraded or non-standard inputs.

3. **Increased Accessibility and Usability:**
    - Provide a tool that allows non-experts, such as enthusiasts to get to know everything about urban farming and this tool is easily accessible.

4. **Scalable and Extensible Framework:**
    - Design the system to support additional languages and scripts in the future, with the potential for integration with larger digitization projects.

The project objectives align with the overarching goal of utilizing technology to help people with the concept of urban farming. By addressing both technical and societal needs, the system aims to not only solve practical challenges but also contribute to the broader mission of making urban farming accessible to a wider audience, ensuring that valuable cultural heritage is preserved for future generations and can be easily accessed for research, education, and personal enrichment.

2)Literature Review

Vertical Farming in Urban Planning: Vertical farming addresses food supply issues due to climate change and urbanization but is not yet integrated into urban planning. A SWOT analysis offers conditions and recommendations for its successful implementation.

Social Benefits of Urban Agriculture: A review of 272 studies highlights urban agriculture's positive impacts on community cohesion, health, and food security, though more research is needed on its educational and economic benefits.

Perception and Acceptance of Urban Agriculture: Residents' positive perceptions of urban farming, particularly its environmental and social benefits, increase acceptance, emphasizing the importance of understanding these perceptions for successful adoption.

Investments and Challenges in Urban Agriculture: Urban agriculture offers benefits like reduced environmental impact but faces challenges, including labor and resource scarcity. Case studies suggest advantages when replacing other land uses, though economic and nutritional impacts remain unclear.

Multifunctionality of Urban Agriculture: Urban agriculture plays a multifunctional role in addressing urban challenges, with developed countries focusing on social, health, and educational benefits. More research is needed to better integrate it into urban planning, especially considering socio-economic contexts.

Urban Farming: A Comprehensive Approach to Sustainable Urban Development

Urban farming presents a sustainable solution to food supply challenges in rapidly urbanizing areas affected by climate change. Vertical farming, a promising innovation, remains underutilized in urban planning but could thrive with strategic SWOT analysis for implementation. Beyond environmental benefits, urban agriculture fosters community cohesion, improves health, and enhances food security, as highlighted in extensive studies.

2.1 NLP Techniques for context handling

Aspect	LSTM	RAG	GRAPH-RAG
Context Handling	Uses Sequence-to-Sequence models and attention mechanisms for contextual understanding.	Retrieves and fuses context-relevant documents for accurate response generation.	Represents context using graphs to capture entity relationships and ensure in-depth understanding.
Memory Retention	Inherently maintains short-term and long-term dependencies.	Uses retrieval-based memory to fetch relevant documents dynamically as required.	Graph structures act as persistent memory by storing and querying relationships between entities.
Multi-hop Reasoning	Chains multiple LSTM layers to process sequential reasoning over complex tasks.	Implements multi-hop retrieval to sequentially gather context for reasoning.	Explores graph paths to infer logical chains of relationships or concepts relevant to the query.
Scalability	Struggles with long input sequences, requiring sliding windows or truncation.	Scales well with large datasets by retrieving only relevant portions of the knowledge base.	Scalable with large, interconnected knowledge graphs, supporting complex queries efficiently.
Domain Adaptability	Requires significant retraining or fine-tuning for domain-specific use cases.	Fine-tuning retriever and generator components enhances domain-specific retrieval and generation.	Adding domain-specific nodes and edges enriches graph utility for specific applications.
Noise Handling	Sensitive to input noise; preprocessing is critical for performance.	Handles noisy input by retrieving supporting documents to clarify ambiguous queries.	Graph-based reasoning mitigates noise by analyzing relationships and focusing on relevant nodes.

Table 2.1 – Comparison between Rag, LSTM, Graph-Rag using some aspects.

3)Work Done

The system design ensures effective utilization of resources using a modular architecture, it integrates an e-commerce-based website along with a Graph-RAG-based chatbot in buying and selling crops produced though urban farming on a marketplace. It will help urban farmers decide what's best for them in terms of the resources and the budget they have.

3.1 Proposed System Architecture

1. Cleaning the Document:

Entity and Relationship Normalization:

- Extract and clean entities (e.g., names, dates, locations) to standardize them, ensuring consistent representation in the graph, which helps in forming accurate connections between nodes.

Noise Removal for Accurate Graph Construction:

- Remove irrelevant text, such as advertisements, formatting artifacts, or non-essential information, to focus on core content, ensuring that the relationships in the graph remain meaningful and contextually relevant.

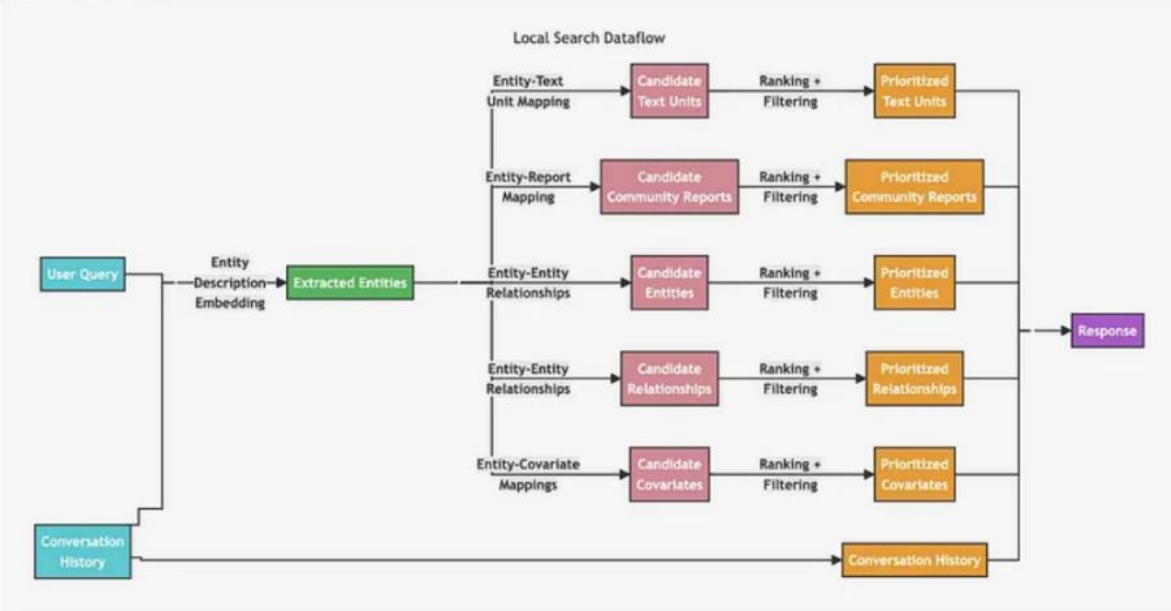

Fig 3.1 - Local Search Dataflow

2. Retrieve Context from Data:

Retrieve prior interactions from memory (Flask session or LangChain memory)

Session-Based Retrieval:

- Use Flask sessions to store and retrieve prior interactions, enabling seamless continuation of conversations without data loss.

LangChain Memory Integration:

- Leverage LangChain's memory features to maintain context across user queries, ensuring responses are informed by previous interactions.

Format the chat history into a context string:

- Contextual Formatting: Convert the chat history into a structured string that integrates prior interactions, ensuring the model has a clear understanding of the conversation's flow.

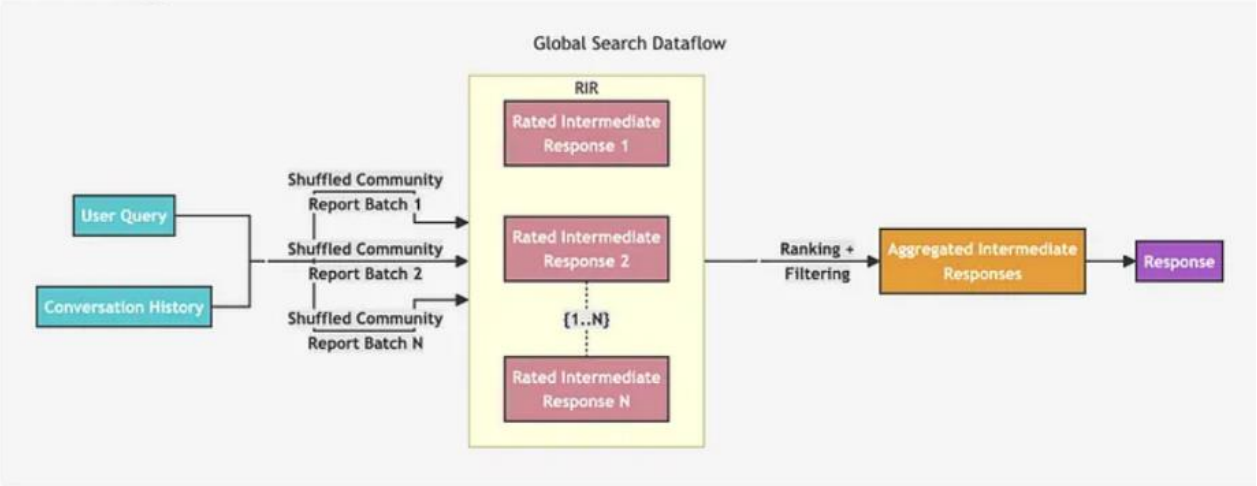

Fig 3.2 – Global Search Dataflow

3. Retrieve Relevant Data:

Query Neo4j for structured data.

Graph-Based Data Retrieval: Use Neo4j's query capabilities to fetch relevant, structured data from the graph database, enhancing the model's response accuracy with context-specific information.

- Perform similarity search using vector embeddings for unstructured data.

Vector-Based Similarity Search: Utilize vector embeddings to perform similarity searches, identifying and retrieving relevant unstructured data based on semantic meaning rather than exact keyword matching.

4. Generate Response:

Combine the retrieved data into the LangChain prompt.

- Consolidate the retrieved data from Neo4j and vector search into a cohesive LangChain prompt, ensuring the context is rich and relevant for the model.

Pass the prompt to the OpenAI API for generating a response.

- Pass the constructed prompt to the OpenAI API, leveraging its capabilities to generate accurate and context-aware responses based on the provided information.

5. Return Answer:

Save the user query and Chatbot response to the context

- Save the user query and chatbot response to the context for maintaining conversation continuity and enhancing future interactions.

Return the Chatbot-generated answer as JSON.

- Return the chatbot-generated answer in JSON format, ensuring structured and easily accessible output for further processing or integration.

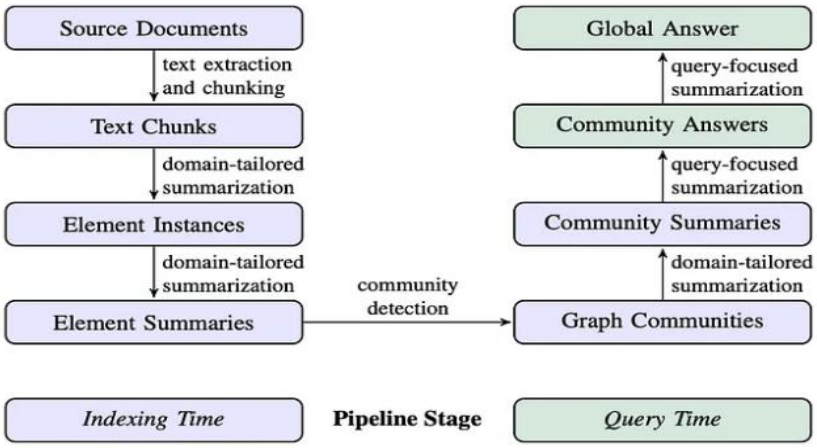

Fig 3.3 - Graph-Rag Pipeline

DATA FLOW DIAGRAM:

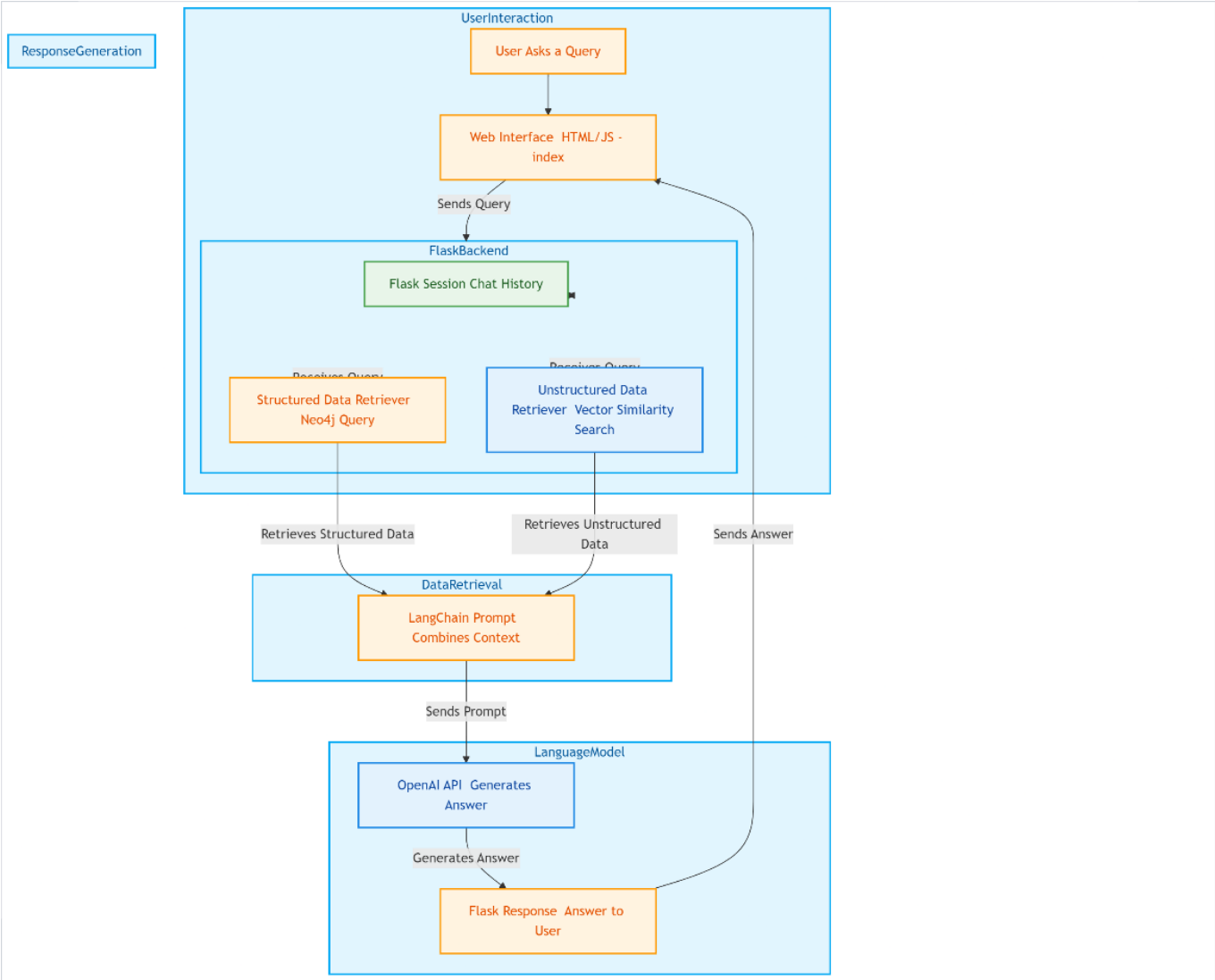

Fig 3.4 - Data Flow Diagram

3.2 Chatbot Module

The Chatbot Module consists of a **Graph-based RAG (Retriever-Augmented Generation) chatbot** which leverages both a **retrieval** mechanism and **generation capabilities** to improve the quality and relevance of responses in a conversation and help non-technical people too through just knowing English and asking questions about a certain topic.

3.2.1 Neo4j Graph Database

The first step in the creation of Neo4j database. These documents serve as the foundation for the entire extraction process. The quality of the documents significantly impact the effectiveness of the context and question-answering.

1. Create a graph schema for storing structured data:

1) **Define Nodes and Labels**:

- Identify the key entities (e.g., Users, Products, Orders) and define them as nodes with appropriate labels, ensuring each entity has unique attributes (e.g., name, ID, timestamp).

2) **Establish Relationships:**

- Create relationships (e.g., "purchased", "owns") between the nodes to represent how entities interact with each other, adding properties to the relationships where necessary (e.g., purchase date, quantity).

3) **Attribute Assignment:**

- Assign relevant attributes to both nodes and relationships (e.g., price for a Product node, amount for a "purchased" relationship), ensuring these attributes are easily queryable and indexable.

4) **Define Indexes and Constraints**:

- Set up indexes on frequently queried attributes (e.g., product ID, user name) and establish constraints (e.g., uniqueness constraints on node IDs) to improve query performance and data integrity.

3.2.2 Vector Store for Unstructured Data

2. Use Neo4jVector to manage embeddings for semantic search:

- Neo4jVector enables **hybrid search** by combining **structural graph relationships** (e.g., parent-child, hierarchical links) with **semantic similarities** derived from embeddings. This dual approach ensures both **contextual relevance** and **logical connections**, making search results more accurate and context-aware.
- Neo4jVector allows the integration of **multi-dimensional embeddings** into graph nodes or edges, preserving rich contextual information. By leveraging graph traversal alongside similarity measures (e.g., cosine similarity), it facilitates **multi-hop reasoning** to uncover hidden insights and relationships.

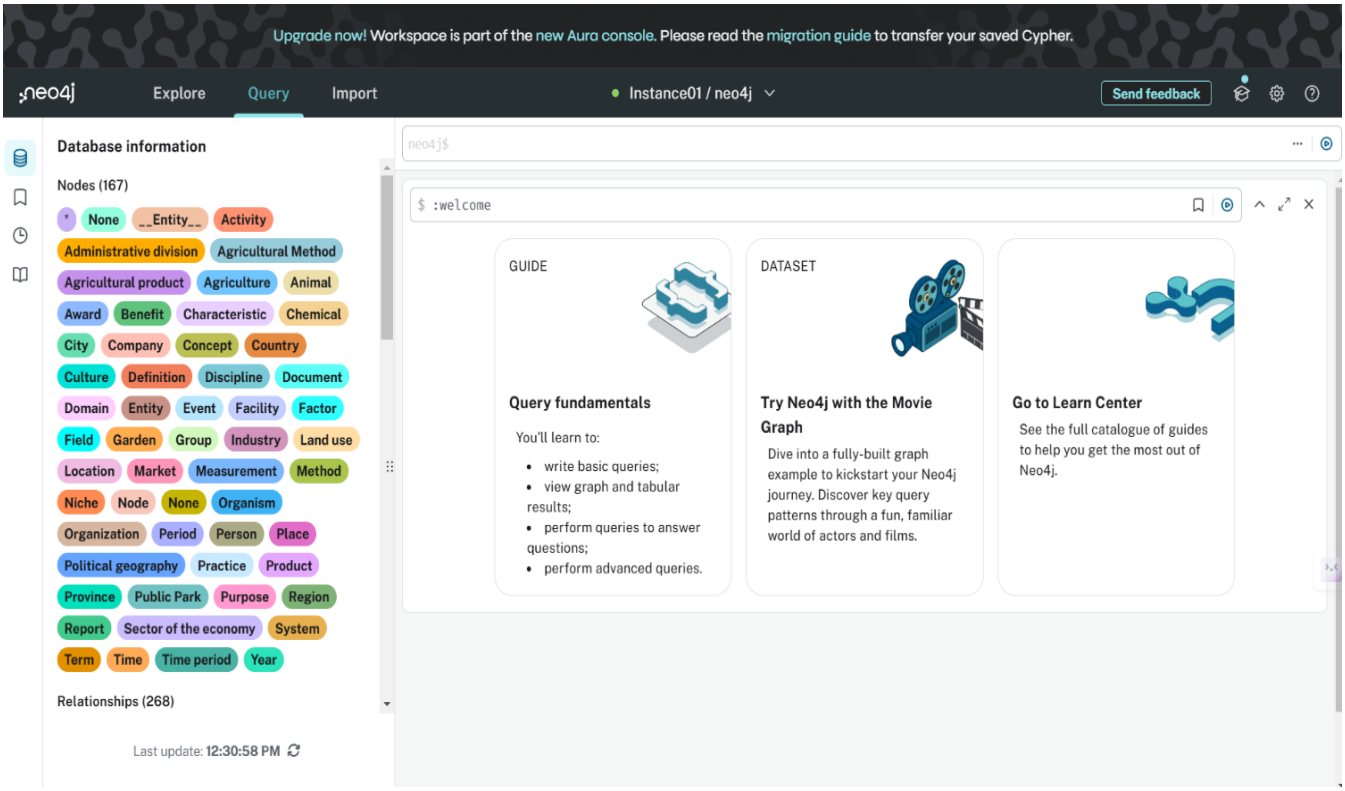

Fig 3.5 - Nodes in Neo4j.

3. Index the text fields from Document nodes in Neo4j for similarity search:

- By indexing text fields within document nodes, Neo4j enables fast and efficient retrieval of relevant documents based on semantic similarity. This indexing optimizes query execution by leveraging precomputed embeddings or textual properties, ensuring low-latency searches even in large-scale datasets.
- Indexed text fields facilitate fine-grained similarity matching by integrating embeddings with advanced similarity metrics (e.g., cosine similarity). This allows for context-aware search results that capture both syntactic and semantic nuances.
- Indexing text fields in Neo4j ensures that scalability is maintained for massive document collections. Neo4j's ability to work with distributed systems and parallel processing ensures that similarity search remains robust and performant as the number of document nodes increases.

3.2.3 LangChain Framework

1. Define a prompt template to guide the language model on how to answer questions.

- A well-defined prompt template provides **clear guidance** to the language model, ensuring it generates **contextually accurate and relevant answers**. By structuring the prompt to include instructions, example responses, or constraints, the model's performance is optimized for domain-specific tasks.

- LangChain’s runnable utilities streamline the process by **orchestrating and managing the pipeline**, enabling seamless integration of prompt templates with other components like retrievers, memory modules, and post-processors. This ensures a **modular and maintainable workflow** for complex systems.

3.2.4 Integrate Context Handling

1. Add Memory to Track Conversations

- Use LangChain's ConversationBufferMemory to maintain context between user queries and AI responses
- **Seamless Context Retention**: LangChain's ConversationBufferMemory enables the system to maintain a persistent understanding of prior interactions, ensuring more coherent and context-aware responses across multi-turn conversations.

- Alternatively, use Flask sessions or a database (e.g., Neo4j) to store and retrieve chat history. Example: ConversationBufferMemory Integration.

- **Flexible Context Storage**: By leveraging Flask sessions or databases like Neo4j, chat history can be efficiently stored and retrieved, enabling scalable and persistent conversation tracking for enhanced user experiences.

3.2.5 Flask Backend

1. Create a REST API with endpoints to handle user queries and return answers

- **Dynamic Query Handling**: A REST API with dedicated endpoints streamlines the process of managing user queries and delivering precise, real-time answers efficiently.

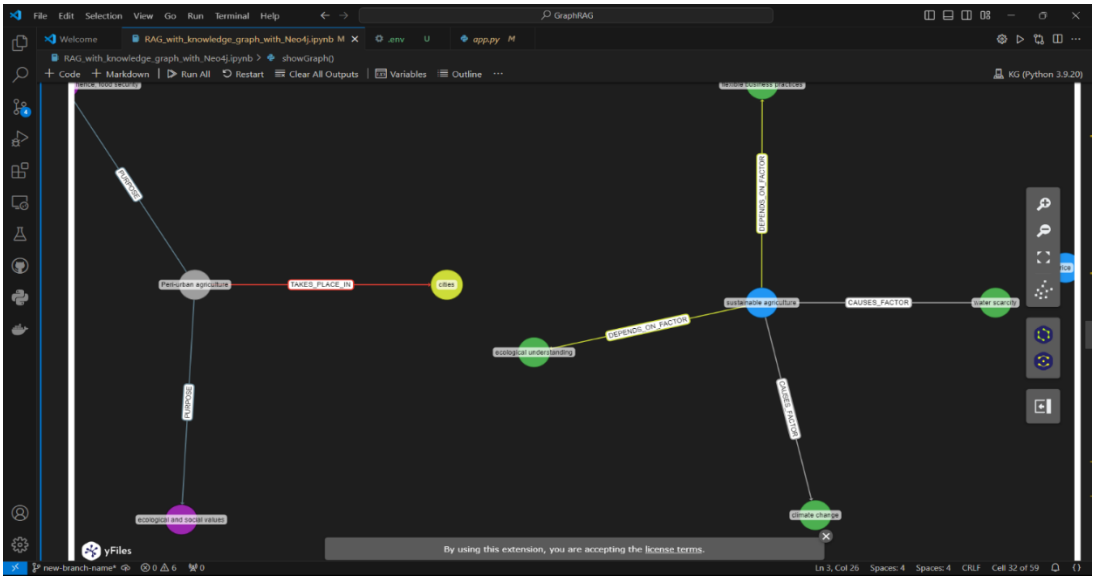

Fig 3.6 - Graph

4)Implementation Details

4.1 Development Tools and Technologies

1. **Programming Languages:**
    - **Python**: Chosen for versatility, machine learning support, and extensive libraries for text and image processing
2. **Libraries and Frameworks:**
    - **Langchain:**
        - Langchain_OpenAI: Provides tools and wrappers specifically for interacting with OpenAI's models (e.g., GPT-3, GPT-4, etc.)
        - Langchain_Community: Community-contributed components that extend LangChain's functionality.
    - **Machine Learning Frameworks:**
        - Tiktoken: Faster Tokeniser than Open-Source Models
    - **Connection Processing Libraries:**
        - Neo4jGraph: - Connection Neo4j with Graph-RAG
        - yfiles_jupyter_graphs:- To plot graphs in jupyter notebooks
    - **Web Development:**
        - Flask: Web Framework to create exceptional websites
    - **Other Libraries:**
        - **NumPy:** For handling large datasets and numerical operations, NumPy is used throughout the system.

4.2 Technical Specifications

1. Hardware Requirements:

- **CPU**: Multi-core processor (Intel i7/AMD Ryzen 7)
- **GPU**: NVIDIA RTX 3050 or higher for deep learning
- **RAM**: 16 GB minimum
- **Storage**: 500 GB SSD

2. Software Dependencies:

- **OS**: Linux (Ubuntu 20.04), compatible with Windows/macOS
- **Python**: Latest 3.x version
- **Database**: Neo4j (scalable)
- **Web Server**: Apache or Nginx

3. System Configuration:

- RESTful API backend

5) Results and Performance Analysis

5.1 Results

Input Details:

- The input is a prompt to the chatbot:- I want to grow watermelons on my backyard can you help me?

- Example:

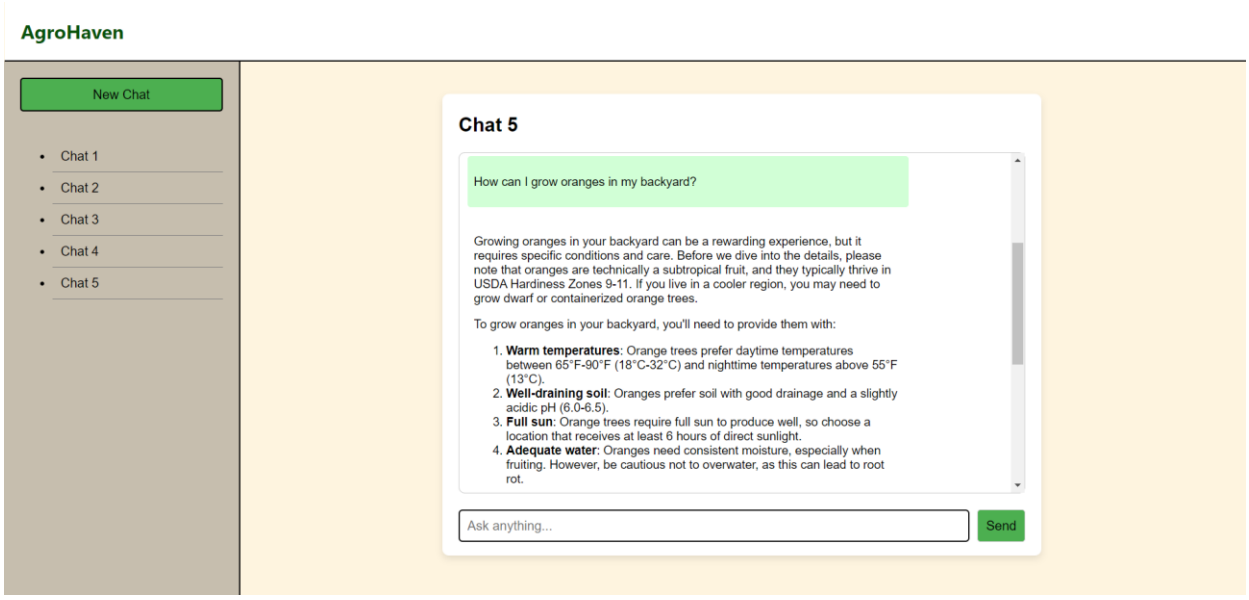

Fig 5.1 Demo1

- Input: How can I grow oranges in my backyard?

- Output: As shown in the figure above

- Similarly, here are some more input prompts

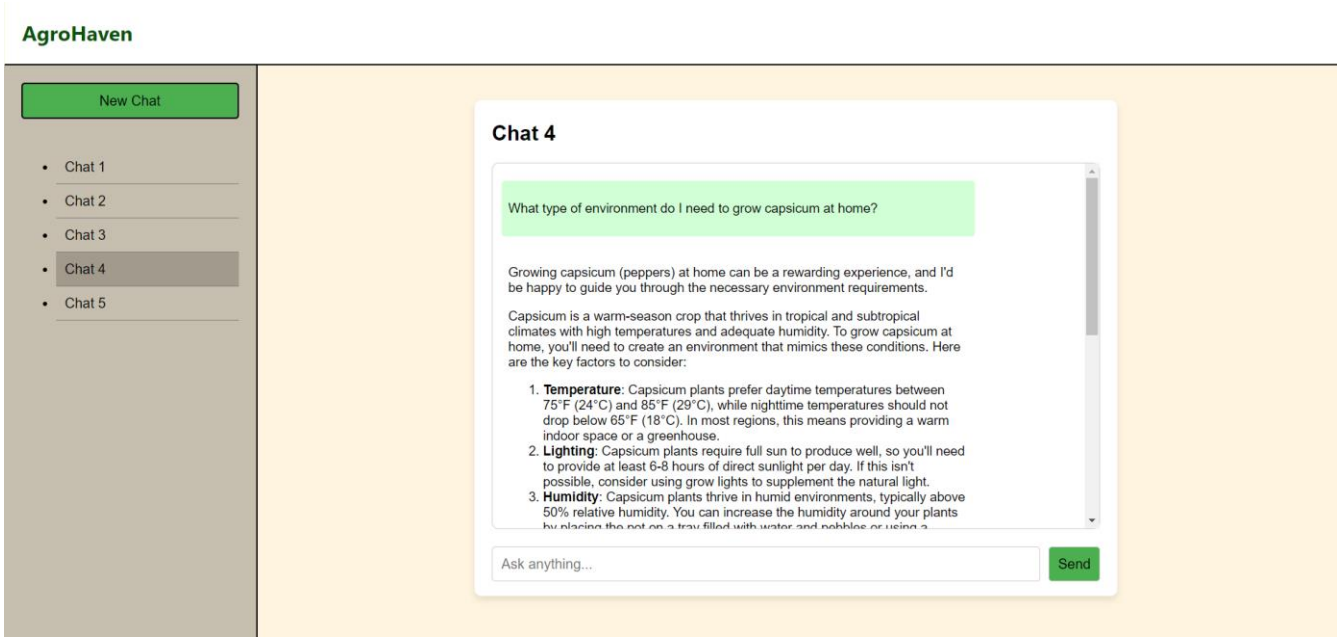

Fig 5.2 Demo2

- **Output Details:**
    - The second image shows the output result of AgroHaven on what type of environment do I need to grow capsicum at home?
    - It demonstrates the successful demonstration of the chatbot response and the e-commerce website and also the exhibition of state-of-the-art results.

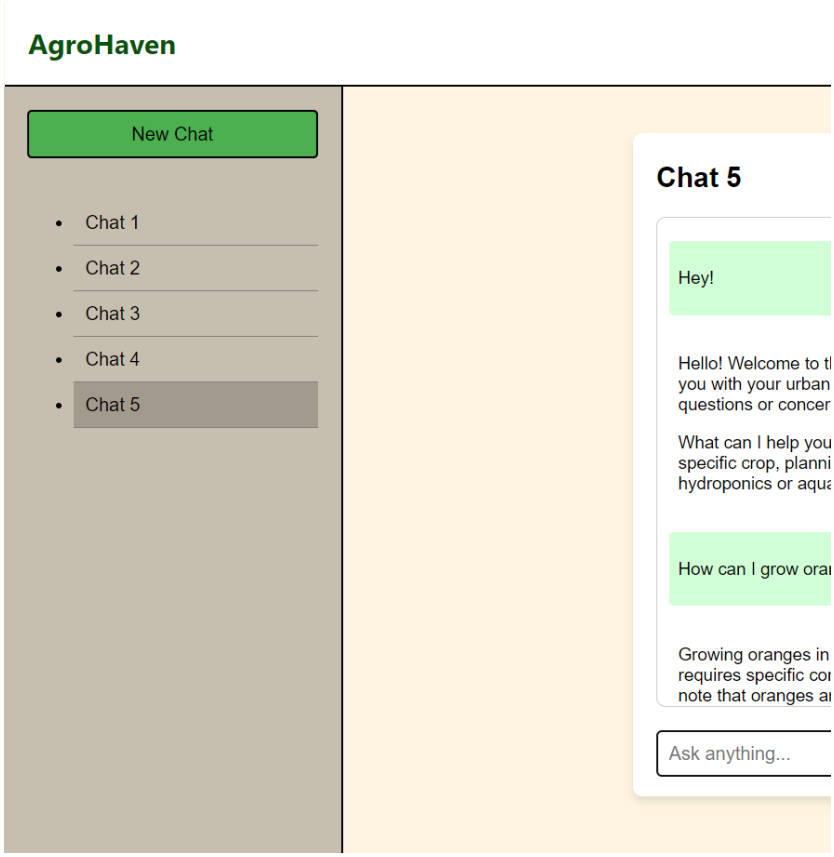

Fig 5.3 History

Highlight: - Here, the chat history is also stored along with the prompts and its responses.

5.2 Performance Metrics:

- BLEU SCORE:
  - A score of 0.42 was secured by the chatbot.

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

- ROUGE SCORE:
  - A score of 0.45 was achieved which is extremely good.

$$\text{ROUGE-L} = \frac{(1+\beta^2) \cdot \text{Precision} \cdot \text{Recall}}{\beta^2 \cdot \text{Precision} + \text{Recall}}$$

- Preservation of Context:
    - Long-term dependencies and contextual accuracy are retained, crucial in handling long prompts and enhancing responses.

5.3 Performance Analysis:

Strengths:

- Context Handling: It handles the context in the prompts exceptionally well, giving extraordinary results.
- Question-Answering Accuracy: The model boasts its accuracy in question-answering related to queries on urban farming.
- Single Platform: All in one platform for everything about Urban Farming.

Challenges:

- Complex Sentence Structures: Handling the complex sentence structures for context handling
- Dependency on the length of documents: The bigger the document, the slower the model gets for processing it into chunks and finding similarity in text vector embeddings.
- Prompt Length: The prompt length should be limited, otherwise the OpenAI API key gets used up and cost increases every input given.

Recommendations:

- Advanced Models: 1) Using Knowledge Graph-RAG for question-answering methods can improve context prediction. 2) Hybrid Approach: Combine rule-based and ML models for specific corrections, especially in multi-word combinations.

6. Conclusion and Future Work

6.1 Project Outcomes

Key Achievements

The project successfully developed an automated system for extracting texts and generating a question-answering chatbot. By integrating effectively converts urban-farming documents into modern, readable text.

Major Contributions

1. **Multimodal Processing Pipeline**: End-to-end solution combining advanced Graph-RAG and question-answering techniques
2. **Context Handling**: Developed methods for handling context of the input, and improve user-experience
3. **Chat History Restored**: Achieved restoration of chat history even after creating new chats and after some latency in responses.

6.2 Limitations

Current System Constraints:

- Limited language and script support
- High dependency on input length
- Slow processing of large documents

Areas Needing Improvement:

- Latency time of responses
- Handling linguistic ambiguities
- Achieving real-time processing capabilities

6.3 Future Research Directions

Potential Extensions:

- Multilingual Chatbot including various dialects also.
- Automation and Robotics: Increased use of robots for planting, harvesting, and maintaining crops will reduce labor costs and increase efficiency.
- Recommendation system to suggest which crops are viable for growing.
- Reduce the latency time of responses in the chatbot.
- Increased government support in terms of subsidies, grants, and favorable policies will encourage the growth of urban farming initiatives, making it a more viable and accessible option for urban populations.
- IoT Integration for real-time price prediction: Smart sensors will provide real-time data on temperature, humidity, soil conditions, and plant health, enabling precision agriculture in urban settings.

The project AgroHaven has demonstrated significant potential in serving and providing resourceful information for Urban Farmers . Future research will focus on overcoming current limitations, extending language support, and integrating more sophisticated machine learning models to enhance the system's scalability and efficiency.

7) References

This section provides a comprehensive list of references, including academic papers, technical documentation, and online resources, that have been consulted or cited throughout the development of the **Urban Farming** All references are formatted according to the **APA** citation style to ensure consistency and academic rigor.

Academic Papers:

1. Specht, K.; Sanyé-Mengual, E. Risks in urban rooftop agriculture: Assessing stakeholders' perceptions to ensure efficient policymaking. *Environ. Sci. Policy* **2017**, *69*, 13–21.

2. Maassen, A. Heterogeneity of Lock-In and the Role of Strategic Technological Interventions in Urban Infrastructure Transformations. *Eur. Plan. Stud.* **2012**, *20*, 441–460.

3. Carolan, M. Urban Farming Is Going High Tech. *J. Am. Plan. Assoc.* **2020**, *86*, 47–59.

4. Engler, N.; Krarti, M. Review of energy efficiency in controlled environment agriculture. *Renew. Sustain. Energy Rev.* **2021**, *141*, 110786.

5. Petrovics, D.; Giezen, M. Planning for sustainable urban food systems: An analysis of the up-scaling potential of vertical farming. *J. Environ. Plan. Manag.* **2021**, *65*, 785–808.

6. Sonnino, R. Food system transformation: Urban perspectives. *Cities* **2023**, *134*, 104164.

Technical Documentation:

1. LangChain : https://langchain.com/
2. Neo4j DataBase : https://neo4j.com/product/neo4j-graph-database
3. Flask: https://flask.palletsprojects.com/en/stable/

Online Resources:

1. Wikipedia Contributors: A collaborative platform for comprehensive knowledge.
2. Medium: A hub for insightful articles and ideas.
3. The Internet Archive: A digital library preserving Farming content.
4. GitHub Contributors: A community driving open-source innovation.

8) Appendices

The Appendices section provides supplementary materials related to the implementation of the m. It includes the complete source code, additional experimental results, and a user manual for system installation, usage, and troubleshooting.

8.1 Source Code

1. Complete implementation code
2. Key algorithm implementations

8.2 Additional Experimental Results

Dataset Preparation:

- The model was trained using a combination of structured data from a **Neo4j graph database** and unstructured documents, including historical texts and digitized records.
- The Neo4j database provided interconnected nodes representing entities (e.g., authors, events, or topics) and relationships, enabling efficient context retrieval.

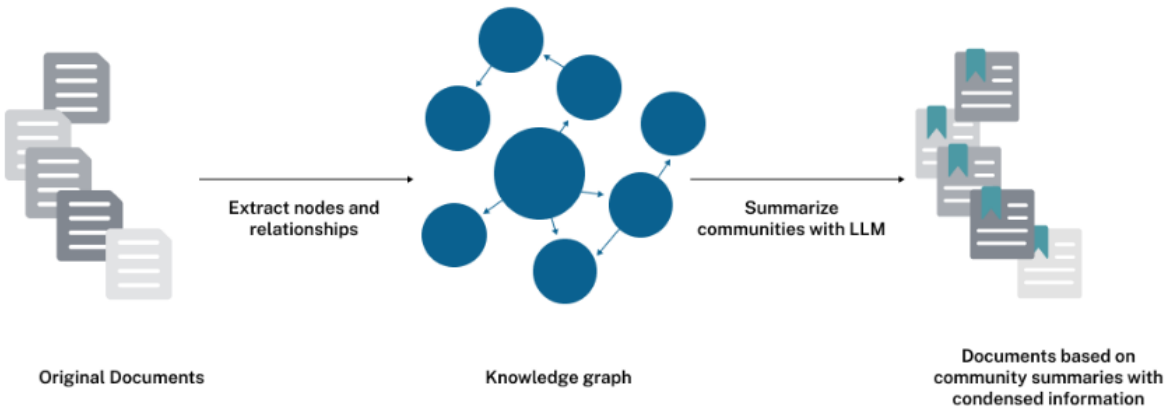

Fig.8.1 Showing Node Extraction and Summarization