## A

Minor Project Report

On

### LLM CHATBOT FOR PUC BY USING RAG

#### Submitted to

**RAJIV GHANDHI UNIVERSITY OF KNOWLEDGE AND TECHNOLOGIES RK VALLEY**

**Accredited by ‘NAAC’ with ‘B+’ Grade**

On completion of Minor project

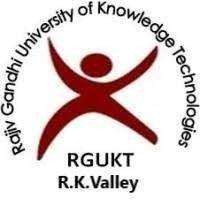
#### Submitted by

#### K. SATHEESH-R190532

**K. THRIVENI-R190533**

Under the Guidance of

**Dr. Ratna kumari Challa**



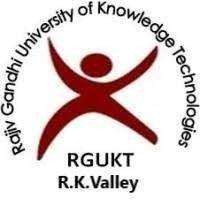
#### DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

**(Catering the Educational Needs of Gifted Rural Youth of AP)**

**R.K Valley, Vempalli(M), Kadapa (Dist.) 516330**

**2023 – 2024**

#### RAJIV GHANDHI UNIVERSITY OF KNOWLEDGE AND TECHNOLOGIES RK VALLEY

**Accredited by ‘NAAC’ with ‘B+’ Grade**

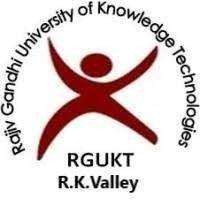
# DECLARATION

I hereby declare that the project report entitled "**LLM CHATBOT FOR PUC BY USING RAG** " submitted to the Department of COMPUTER SCIENCE AND ENGINEERING in partial fulfilment of requirements for the award of the degree of BACHELOR OF TECHNOLOGY. This project is the result of my own team effort and that it has not been submitted to any other University or Institution for the award of any degree or diploma other than specified above.

**K. SATHEESH (R190532)**

**K. THRIVENI (R190533)**

#### RAJIV GHANDHI UNIVERSITY OF KNOWLEDGE AND TECHNOLOGIES RK VALLEY

**Accredited by ‘NAAC’ with ‘B+’ Grade**

# CERTIFICATE

This is to certify that the project work titled **“LLM CHATBOT FOR PUC BY USING RAG”** is a bonafied project work submitted by **K. SATHEESH–R190532** AND **K. THRIVENI-R190533** in the department of COMPUTER SCIENCE AND ENGINEERING in partial fulfilment of requirements for the award of degree of Bachelor of Technology in Computer Science and Engineering for the year 2023- 2024 carried out the work under the supervision.

**Signature of Project Guide**

**Dr.CH Ratna Kumari**

**Signature of the HOD**

**Dr. CH Ratna Kumari**

Head of the Department

Computer Science and Engineering RGUKT RKValley

# ACKNOWLEDGEMENT

I have great pleasure in expressing our hearty thanks to our beloved Director **Prof. A V S S KUMARA SWAMI GUPTA** for spending his valuable time with us to complete this project.

We would like to express our gratefulness and sincere thanks to **Dr. CH RATNA KUMARI** Head of the Department of COMPUTER SCIENCE AND ENGINEERING, for his kind help and encouragement during our study and in the successful completion of the project work.

We are thankful to our guide **Dr. CH RATNA KUMARI** for his valuable guidance and encouragement. His helping attitude and suggestions have helped us in the successful completion of the project.

Successful completion of any project cannot be done without proper support and encouragement. We sincerely thanks to the **Management** for providing all the necessary facilities during the course of study.

We would like to thank our parents and friends, who have the greatest contributions in all our achievements, for the great care and blessings in making us successful in all our endeavours.

**WITH SINCERE REGARDS**

**K. SATHEESH(R190532)**

**K. THRIVENI(R190533)**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **INDEX** | **PG.NO** |
| 1 | Abstract | 6 |
| 2 | List of Figures | 7 |
| 3 | Introduction | 8-9 |
| 4 | Literature Review& Technologies used | 9-16 |
| 5 | Methodology  Modules (1-10) | 17-29 |
| 6 | Result & Discussion | 30-33 |
| 7 | Future Enhancement | 34-36 |
| 8 | Conclusion | 37 |
| 9 | References | 38 |

# ABSTRACT

In recent years, chatbots have emerged as a revolutionary tool in various sectors, leveraging advances in artificial intelligence (AI) and natural language processing (NLP). This project presents the development of a chatbot designed specifically for first and second-year biology intermediate students at our college. The motivation behind this project stems from the need to provide students with a reliable and accessible platform for addressing their academic queries. The chatbot aims to facilitate students in resolving their doubts efficiently, thereby enhancing their learning experience and academic performance. Leveraging cutting-edge generative AI technologies, including large language models (LLMs) and Retrieval-Augmented Generation (RAG), by using RAG It retrieves relevant documents and pieces of information from a vast corpus and uses this data to generate precise and contextually appropriate responses and also it enhances the chatbot's capabilities. To manage and access the vast amount of academic content. MongoDB Atlas is employed as the knowledge base. MongoDB Atlas provides a scalable and flexible platform to store and retrieve information efficiently. The implications of this project for the college students are substantial. By providing an accessible and efficient academic support tool the students can self-directed to learning, enabling students to explore and understand topics at their own pace.

The value of this project lies in its potential to transform the academic support landscape within the college. It serves as a scalable solution that can be extended to other subjects and departments, offering a versatile tool for academic assistance. Furthermore, the integration of advanced AI technologies such as LLM and RAG sets a precedent for future educational innovations, positioning the college at the forefront of educational technology adoption.

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure No** | **TITLE** | **PG No** |
| 1 | MPNET Embedding Architecture | 17 |
| 2 | Database | 22 |
| 3 | Streamlit running in terminal | 30 |
| 4 | interface | 30 |
| 5 | Input/query | 31 |
| 6 | output | 31 |
| 7 | Original content | 32 |
| 8 | Query and Response | 32 |
| 9 | ChatGPT output | 33 |

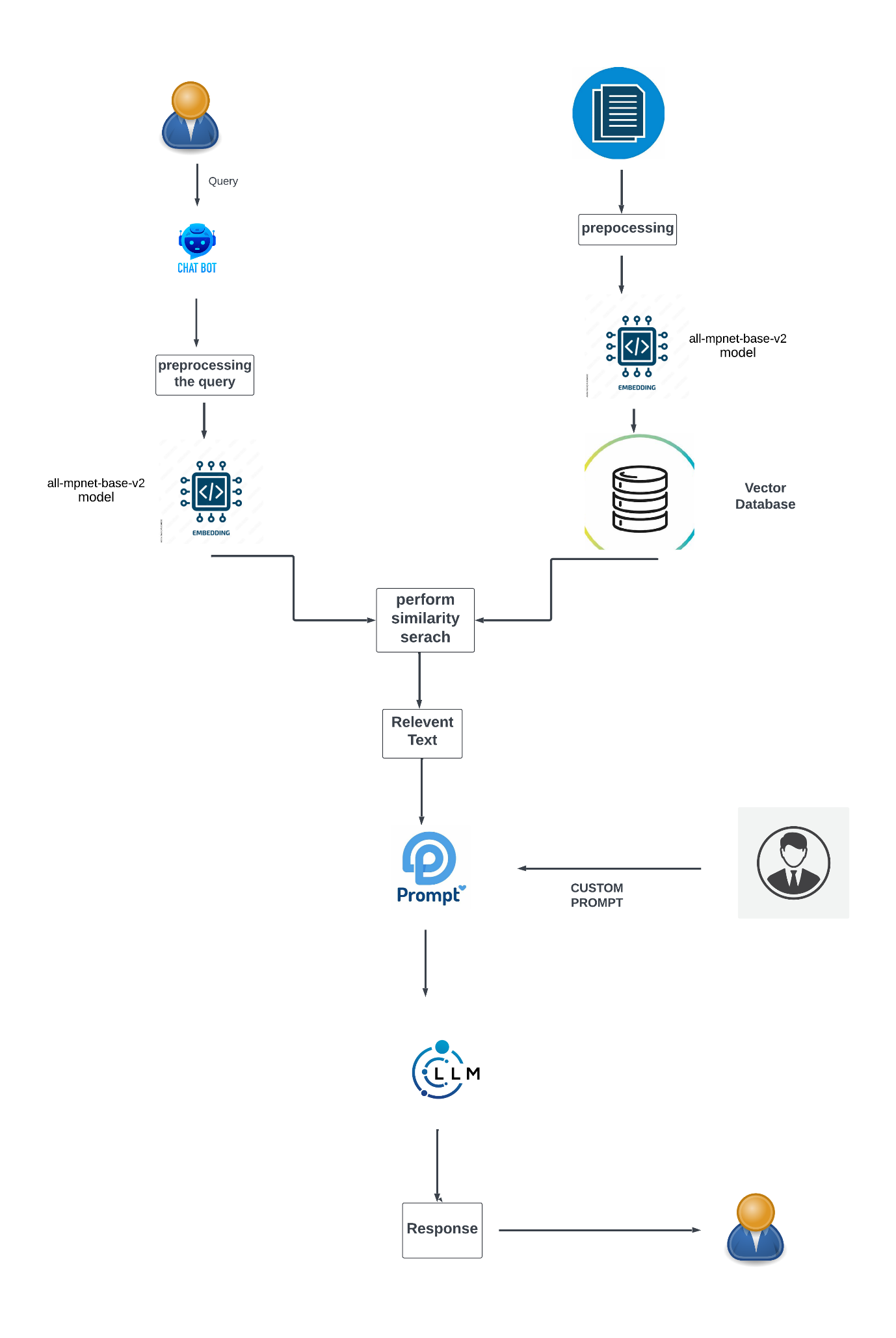
**Chapter-I**

**INTRODUCTION**

In the realm of modern education, the integration of advanced technological solutions has become indispensable for fostering an enriching learning environment. One such innovation is the deployment of chatbots, which harness the power of artificial intelligence (AI) to provide instant and tailored support to students. This project undertakes the development of a sophisticated chatbot tailored specifically for first and second-year biology students at our institution, aimed at addressing their academic queries with unparalleled efficiency. The motivation for this project is to overcome the limitations of traditional educational support, which often cannot provide timely assistance to students outside of classroom hours. By using advanced AI technologies, including Large Language Models (LLMs) and Retrieval-Augmented Generation (RAG) tools, this chatbot is designed to give accurate and contextually relevant answers, thus improving the learning experience. The objectives of this project are clear and ambitious. We aim to develop a robust, scalable platform that can handle a diverse array of queries, ranging from simple factual questions to complex conceptual clarifications.

The chatbot is trained on a comprehensive dataset encompassing NCERT books and supplementary academic resources, ensuring an extensive coverage of the biology curriculum. Additionally, the integration of MongoDB Atlas as the knowledge base provides a scalable and flexible framework for efficient data retrieval and management. This project not only aims to improve the academic support available to biology students but also serves as a model for future educational innovations. The implications of this project extend beyond immediate academic benefits, positioning our college at the forefront of educational technology and innovation.

In summary, this project demonstrates the potential of AI in education, showing how chatbots can create a more responsive and effective learning environment. The following sections of this report will cover the literature review, detailed methodology, implementation process, and key results, leading to a discussion of the project's broader impacts and future possibilities.



**Complete process**

**Chapter-II**

**Literature Review**

The field of artificial intelligence (AI) has seen significant advancements in recent years, with applications spanning across various domains, including education. Chatbots, in particular, have garnered considerable attention for their potential to enhance educational experiences by providing instant and personalized support to students. This literature review explores the existing research and developments in the areas of chatbots, generative AI technologies, and their applications in educational contexts, with a specific focus on Large Language Models (LLMs), Retrieval-Augmented Generation (RAG) tools, and the utilization of MongoDB Atlas as a knowledge base.

**Chatbots in Education**

The deployment of chatbots in educational settings is not a novel concept; however, recent technological advancements have significantly augmented their capabilities and efficacy. Chatbots have been utilized to assist with administrative tasks, provide language learning support, and facilitate tutoring in various subjects. According to Winkler and Sollner (2018), chatbots can enhance student engagement and motivation by providing immediate feedback and fostering interactive learning environments. Furthermore, studies by Hill et al. (2015) highlight the effectiveness of chatbots in offering personalized learning experiences tailored to individual student needs.

### Large Language Models (LLMs)

Large Language Models, such as OpenAI's GPT-3, have revolutionized the field of natural language processing (NLP). These models are trained on vast datasets and possess the ability to generate human-like text based on given prompts. Research by Brown et al. (2020) demonstrates the versatility of LLMs in performing a wide array of language tasks, from translation to summarization and question answering. In the context of education, LLMs have shown promise in generating contextually relevant responses and explanations, thereby serving as effective tools for academic support

### Retrieval-Augmented Generation (RAG) Tools

Retrieval-Augmented Generation (RAG) represents a sophisticated approach that combines the strengths of information retrieval and generative modelling. RAG models, as described by Lewis et al. (2020), first retrieve relevant documents from a corpus based on the input query and then generate a response using this retrieved information. This dual mechanism enhances the accuracy and contextual relevance of the responses, making RAG tools particularly valuable in educational applications where precise and contextually appropriate answers are crucial.

### MongoDB Atlas in Educational Applications

The use of robust database solutions is critical for managing the vast amounts of information required for effective educational support systems. MongoDB Atlas database offers scalability, flexibility, and efficient data retrieval capabilities. As noted by Chodorow (2013), MongoDB's schema-less architecture and powerful query capabilities make it an ideal choice for dynamic and data-intensive applications. In educational contexts, MongoDB Atlas has been employed to store and manage diverse educational content, enabling quick and reliable access to information.

### Integration of AI Technologies in Educational Chatbots

The integration of LLMs, RAG tools, and MongoDB Atlas in educational chatbots represents a significant advancement in the field. Studies by Xie et al. (2021) and Rajpurkar et al. (2018) underscore the potential of these technologies to provide accurate and contextually enriched academic support. The combination of advanced AI models with efficient data management systems enhances the chatbot's ability to deliver timely and relevant assistance to students, thereby improving their learning outcomes.

### Implications and Future Directions

The existing literature underscores the transformative potential of AI-driven educational chatbots. The integration of LLMs and RAG tools, supported by robust database solutions like MongoDB Atlas, can significantly augment the capabilities of educational support systems. Future research should focus on refining these technologies to further enhance their accuracy and contextual understanding. Additionally, exploring the application of these advancements across various subjects and educational levels can provide deeper insights into their efficacy and broader impact.

**Technologies Used**

**Mongo Client:**

The MongoDB client library, often referred to as the MongoDB driver, allows applications to interact with a MongoDB database. Here are some key aspects of the MongoDB client library:

### Key Features:

1. **Connection Management**: Manages connections to MongoDB instances, including connection pooling and replica set management.
2. **CRUD Operations**: Supports Create, Read, Update, and Delete operations on the MongoDB database.
3. **Querying**: Allows for complex querying capabilities including filtering, sorting, and aggregation.
4. **Indexing**: Supports the creation and management of indexes to optimize query performance.
5. **Transactions**: Provides support for multi-document ACID transactions (in MongoDB 4.0 and later).
6. **Change Streams**: Enables real-time updates by allowing applications to subscribe to changes in the database.
7. **Grid FS**: Supports storing and retrieving large files that exceed the BSON-document size limit

.**Quote-plus:**

The quote-plus function is part of the urllib.parse module in Python's standard library. It is used to encode strings into a format suitable for inclusion in a URL query string.

### Key Features:

1. **Encoding Special Characters**: Converts special characters into their URL-encoded equivalents. For example, spaces are converted to +, and characters like / are converted to %2F.
2. **Usage in Query Strings**: Particularly useful for encoding the parameters of a URL's query string to ensure they are properly formatted and safely transmitted over HTTP.

**Numpy:**

* NumPy (Numerical Python) is a fundamental package for scientific computing in Python, providing support for arrays, matrices, and a broad collection of mathematical functions to operate on these data structures.
* It is essential for numerical calculations and serves as the foundation for more advanced scientific libraries such as SciPy, Pandas, and Matplotlib. NumPy offers efficient storage and manipulation of large datasets, with operations that are optimized for performance. At its core, NumPy introduces the nd array, a multi-dimensional array object that allows for fast and flexible data manipulation.

**Langchain:**

* LangChain is a Python library designed to facilitate the creation of applications that leverage language models, such as GPT-3. It simplifies the process of building natural language processing (NLP) applications by providing a framework for chaining together various NLP components and tasks.
* LangChain supports integration with popular language models and tools, allowing developers to focus on building high-level functionality without worrying about low-level details.
* It includes utilities for text generation, summarization, question answering, and more, making it ideal for creating chatbots, content generators, and other AI-driven applications. Additionally, LangChain provides a flexible architecture that can be extended and customized to fit specific use cases, enhancing its versatility in the development of sophisticated NLP solutions.

**Hugging face:**

* Hugging Face is a leading open-source library for natural language processing (NLP) that provides state-of-the-art pre-trained models and tools for tasks such as text classification, translation, summarization, and question answering. The library includes the popular Transformers library, which offers easy access to models like BERT, GPT, and T5, facilitating their use in various applications.
* Hugging Face simplifies the deployment and fine-tuning of these models, enabling developers to integrate powerful NLP capabilities into their projects with minimal effort. It supports multiple programming languages and frameworks, enhancing its accessibility and utility. The library also features the Datasets library, which provides a wide range of curated datasets for training and evaluation.
* Hugging Face's vibrant community and comprehensive documentation make it a go-to resource for both beginners and experts in the NLP field.

**Groq:**

* Groq is a technology company focused on developing advanced tensor processing units (TPUs) specifically designed for high-performance computing in artificial intelligence (AI) and machine learning (ML) applications. Their TPUs are engineered to deliver exceptional computational speed and efficiency, aiming to maximize throughput while minimizing latency. Groq's processors are scalable, enabling the creation of powerful computing clusters suitable for extensive AI workloads.
* They are also power-efficient, balancing high performance with low energy consumption. Groq supports a variety of AI and ML frameworks, facilitating seamless integration into existing workflows.
* The company offers GroqWare, a suite of tools and libraries to optimize applications for their TPUs. Groq is becoming a significant player in the AI hardware space, pushing the boundaries of AI and ML performance.

**Streamlit:**

* Streamlit is an open-source Python library that makes it easy to create and share beautiful, custom web applications for machine learning and data science projects. With Streamlit, developers can transform data scripts into interactive web apps in just a few lines of code, without needing extensive web development knowledge.
* The library provides a simple API to add widgets, charts, and interactive elements, allowing users to visualize data and models in real-time. Streamlit automatically detects changes in the code and updates the app live, providing a seamless development experience. It supports integration with popular data science libraries such as Pandas, Matplotlib, and Plotly. Streamlit apps can be deployed on various platforms, making it convenient to share insights and results with others. This library is particularly popular for rapid prototyping and sharing data-driven applications.

**Chapter-III**

**Module 1: Preprocessing the document in the backend**

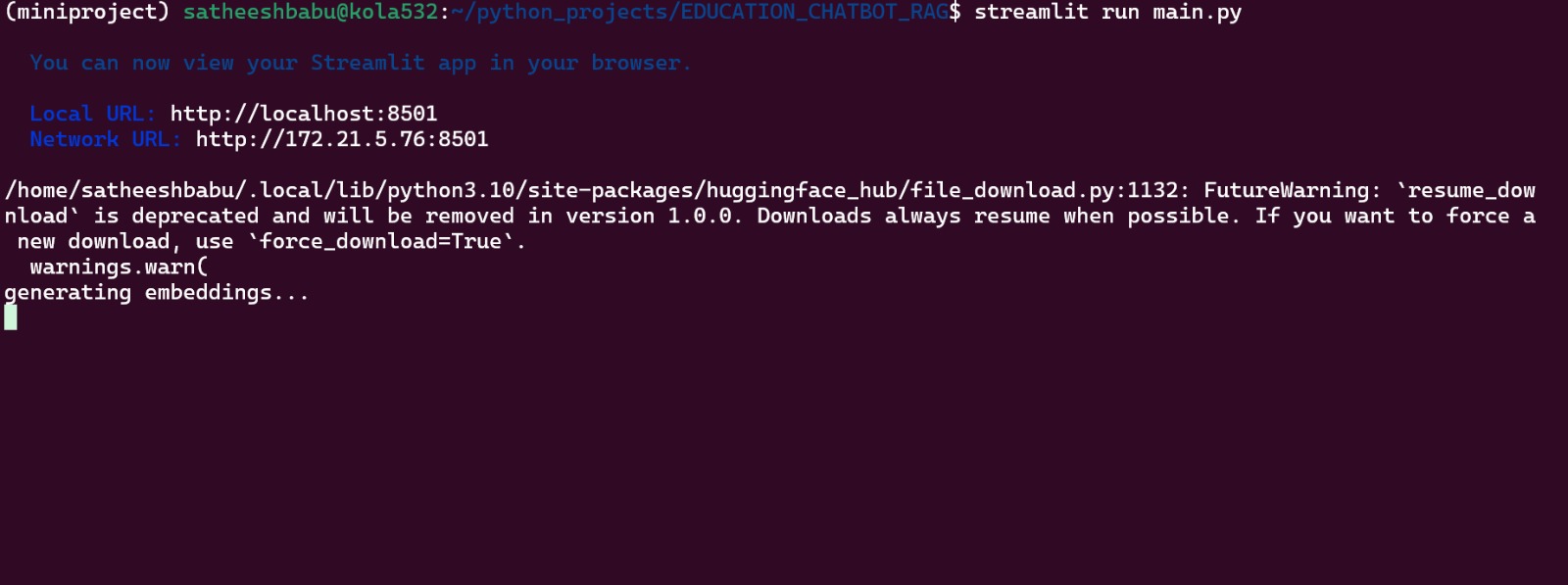
1. **Document and Data Preprocessing**

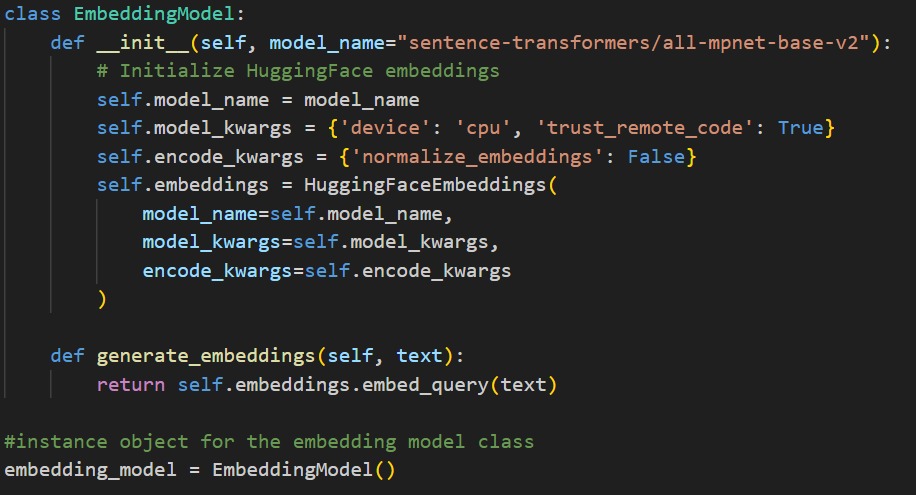
* **Documents**: The process begins with the collection of documents, which contain the relevant information and content needed for the chatbot.
* **Pre-process**: These documents undergo a preprocessing stage where the text is cleaned, normalized, and formatted for further analysis. This may include steps such as removing special characters, correcting types, and tokenizing text into manageable units.

**Module 2: Converting text into vectors**

**Embedding Generation**

* Sentence Transformers convert sentences and paragraphs into dense vector embeddings. These embeddings are high-dimensional vectors that encode the semantic meaning of the text, making them useful for tasks that require understanding the nuances of language.
* Sentence Transformers come with several pre-trained models optimized for different tasks and datasets. The all-mpnet-base-v2 model, for instance, is known for its robust performance in generating high-quality embeddings for various applications
* These models learn to map similar words to nearby points in a high-dimensional space based on their context in a large corpus of text. Embeddings are crucial for tasks such as text classification, sentiment analysis, and information retrieval, as they enable machines to understand and process human language more effectively.
* They also facilitate transfer learning, where pre-trained embeddings can be fine-tuned for specific applications, enhancing performance with less training data





**Document Embedding Generation**:

* All documents and relevant texts are preprocessed and passed through the all-mpnet-base-v2 model to generate dense vector embeddings. These embeddings are stored in a vector database for efficient retrieval.
* After processing the text through transformer layers, Sentence Transformers use pooling layers to aggregate the contextual information into a single fixed-size embedding. Common pooling strategies include mean-pooling, max-pooling, or using the CLS token's representation.
* The output is a document embedding, represented as a vector (e.g., <0.1, 0.4, -0.2, ...>), which can be efficiently stored and retrieved.

**MPNET Embedding Architecture:**

* This is a pre-trained deep learning model used to create numerical representations (embeddings) of text. It analyzes the context of words and captures their semantic meaning and relationships.
* In RAG, the MPNET model takes queries and document chunks as input and generates dense vectors for each. These vectors represent the meaning of the text in a high-dimensional space.

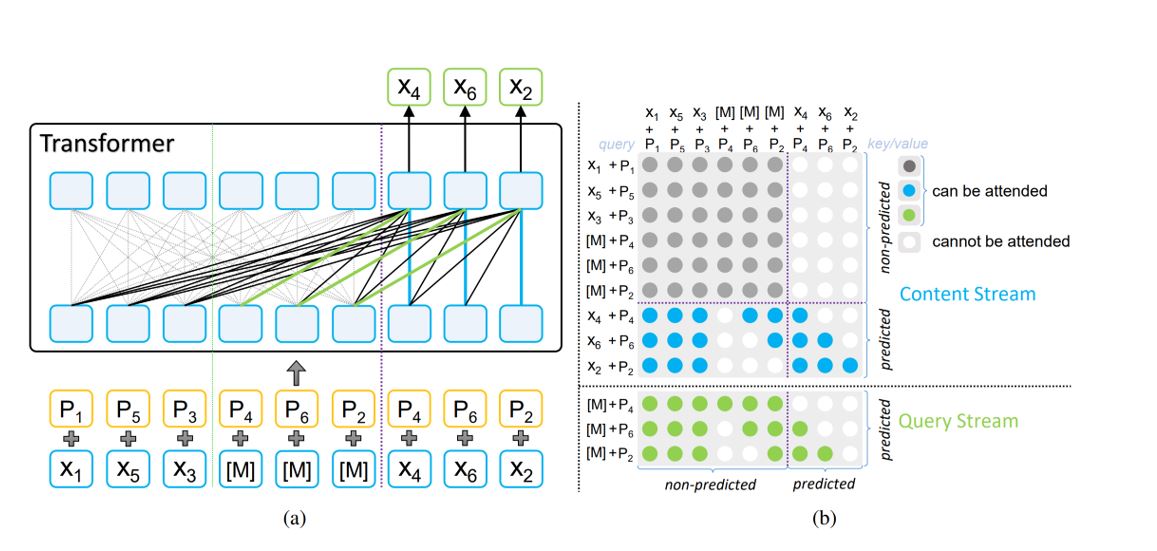


Fig 1- MPNET Embedding Architecture

In the context of the MPNet architecture, the symbols x1, x2, x3, x4, x5, and x6 represent individual tokens or words within an input sentence. These tokens are the fundamental units of text that the model processes. The symbols p1, p2, p3, p4, p5, and p6 represent the positional embeddings associated with each token. Positional embeddings provide information about the location of each token in the sentence, which is crucial for the model to understand the order and relationships between words

**Flow Depicting Arrow Marks**:

The arrow marks in the MPNet architecture diagram illustrate the flow of information within the model during the training process.

**Here's a breakdown of the flow:**

**1.** **Input:** The input sentence is tokenized (split into individual words or sub words) and converted into embeddings (x1, x2,..., xn) along with their corresponding positional embeddings (p1, p2..... pn).

**2. Permutation:** The order of the tokens is randomly permuted (shuffled). This is a key aspect of MPNet, as it helps the model learn bidirectional relationships between words.

**3. Masking:** Some of the tokens in the permuted sequence are masked, meaning they are replaced with a special "[MASK]" token.

**4. Two-Stream Processing:** The masked sequence is then processed by two parallel streams:

* **Content Stream:** This stream receives the entire permuted sequence, including both masked and non-masked tokens. It processes the full context of the sentence.
* **Query Stream:** This stream receives the permuted sequence with masked tokens. It focuses on predicting the masked tokens based on the context provided by the content stream.

**5. Self-Attention**: Both streams use self-attention mechanisms to weigh the importance of different tokens in the sequence. The query stream attends to the non-masked tokens in the content stream to gather information for predicting the masked tokens.

**6. Position Compensation:** The query stream also incorporates positional information to compensate for the fact that it cannot see the masked tokens directly. This helps maintain consistency between pre-training and fine-tuning.

**7. Prediction:** The query stream produces output probabilities for each possible token that could fill the masked positions. The model is trained to maximize the probability of predicting the correct masked tokens.

In essence, the arrows represent the flow of token and positional embeddings through the MPNet model, highlighting the interaction between the content and query streams, the role of self-attention, and the incorporation of positional information for accurate language understanding

**Module 3: Vector database storage**

**Vector databases** are designed to handle high-dimensional vector data, enabling efficient storage, retrieval, and similarity search. MongoDB Atlas, the cloud-based managed database service by MongoDB, has introduced features that support vector data, making it suitable for applications that require handling embeddings and performing similarity searches.

* MongoDB allows the creation of specialized indexes to support fast vector search queries. These indexes can be used to accelerate the performance of similarity search operations.
* Before storing vectors, the data (e.g., documents, images) is preprocessed to extract relevant features or embeddings using models like Sentence Transformers. The output is a high-dimensional vector that represents the semantic content of the data.

**Document Structure**:

* Each document in MongoDB Atlas can include the original data (e.g., text, metadata) and the corresponding vector embedding.
* These documents are inserted into a MongoDB collection. MongoDB’s flexible schema allows storing vectors as arrays within the document, ensuring all relevant information is stored together.

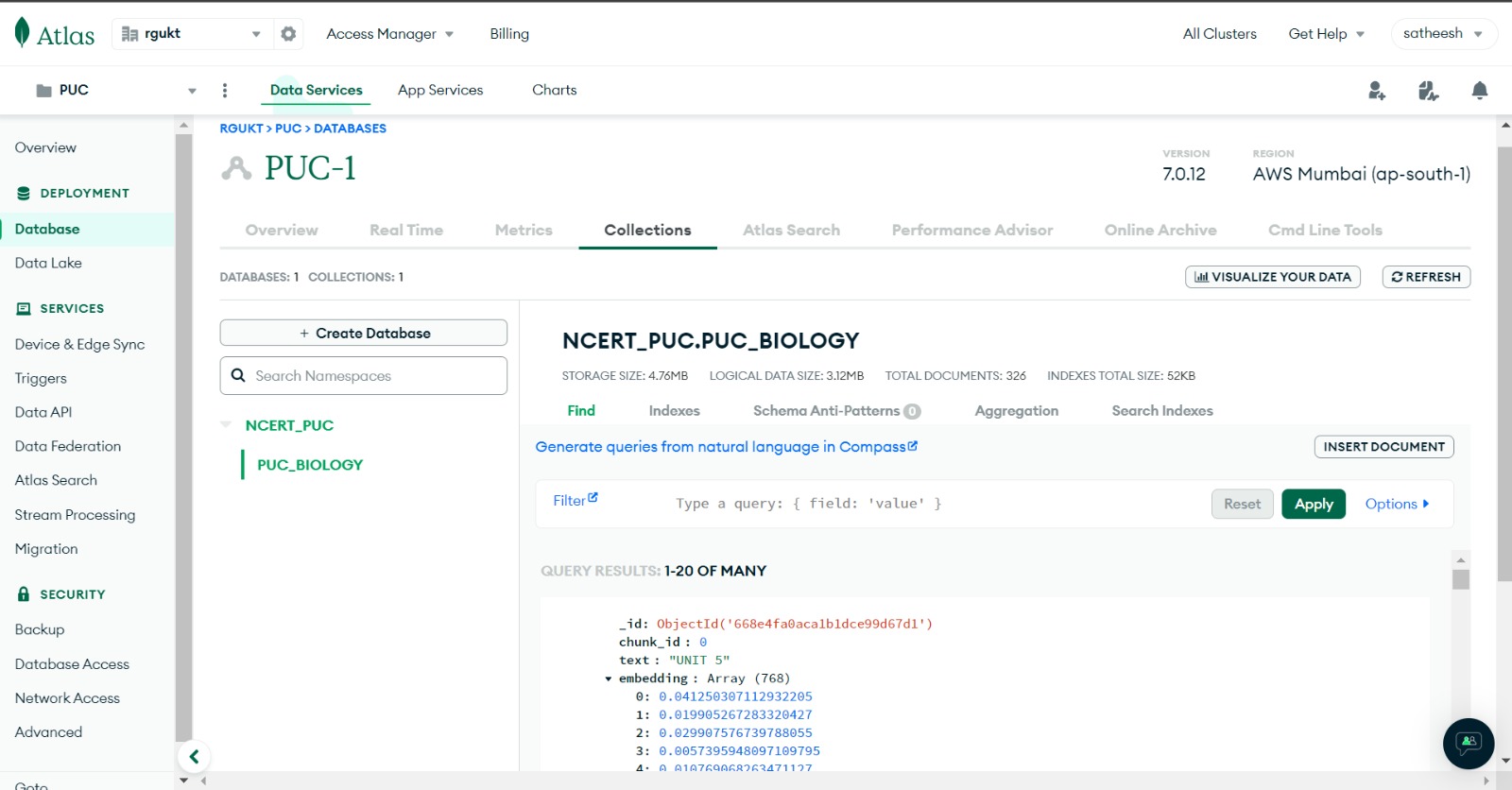


Fig 2-Database

**Chapter-IV**

**Module 4: User Query Submission**

**1.User Interface Interaction**

The user interacts with a web-based interface that allows for seamless query submission. The interface is designed to be intuitive and user-friendly, ensuring that students can easily navigate and utilize the chatbot's features.

* **Text Input Field**: A clearly marked text box where users can type their questions. This field supports natural language input, enabling users to phrase their questions as they would in a conversation.

**2.Query Submission**

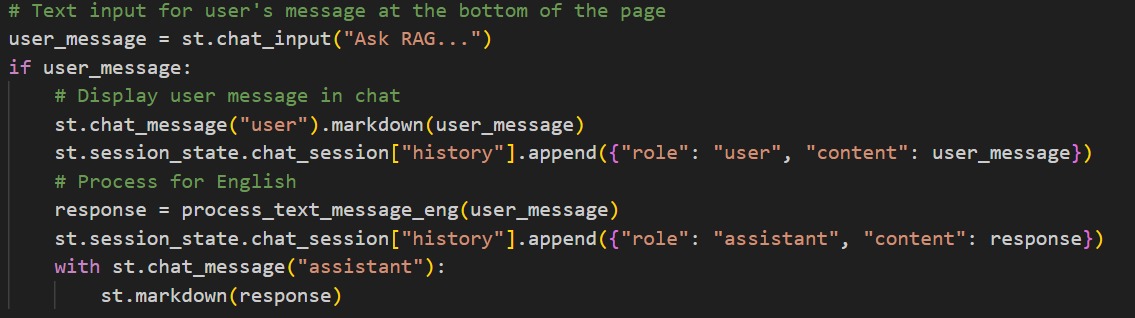
* Once the user has entered their query or uploaded an image, they initiate the submission process by clicking the "Submit" button. The interface captures the query data and sends it to the backend for processing.

**Module 5: Preprocessing the query**

**Data Handling and Preprocessing**

Upon receiving the query, the backend performs initial data handling and preprocessing to ensure the input is in a suitable format for further processing.

* **Text Queries**: The text input is cleaned and tokenized. Cleaning involves removing unnecessary whitespace, correcting common types, and normalizing text to a consistent format. Tokenization breaks the text into smaller units (tokens), which are easier for the AI models to process.



**Module 6**: **Converting text into vectors**

**Query Embedding Generation**:

* When a user submits a query, it is similarly preprocessed and transformed into an embedding using the same all-mpnet-base-v2 model. This ensures that both document and query embeddings are in the same vector space, facilitating accurate similarity comparisons.

**Module 7:** **Retrieving Relevant Information**

# The query embedding is used to search the vector database for relevant documents. The database retrieves documents whose embeddings are most similar to the query embedding, based on cosine similarity

# Semantic and Similarity Techniques

1. **Semantic Embeddings**:

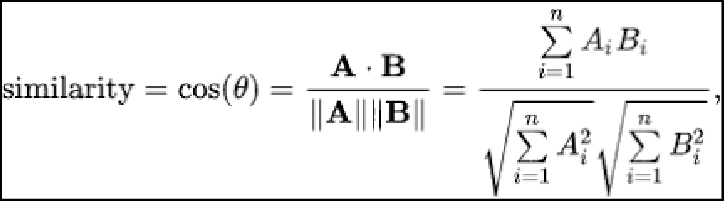
* Semantic embeddings are dense vector representations of text (or other data) that capture the meaning and context of the content. These embeddings are generated using models like Sentence Transformers, which transform input text into high-dimensional vectors.
* The core idea is that semantically similar pieces of text will have similar embeddings, meaning their vectors will be close to each other in the embedding space.

1. **Similarity Search**:

# Similarity search involves comparing the query embedding with embeddings stored in the database to find the closest matches. The measure of "closeness" or similarity can be computed using various mathematical techniques, with cosine similarity being one of the most common.

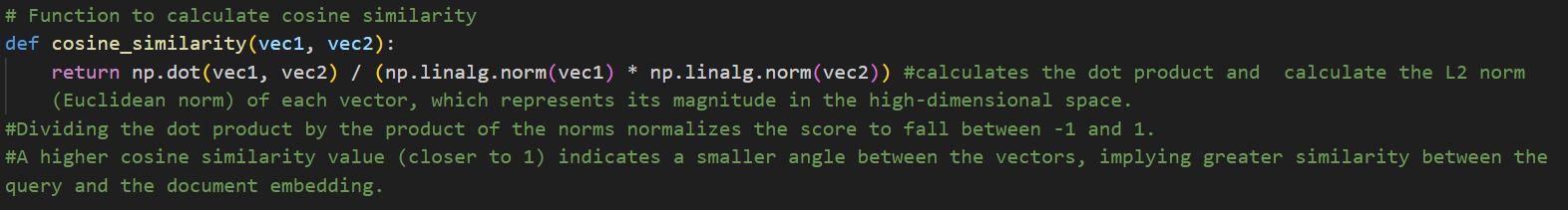
**Cosine Similarity:**

* This is a mathematical metric used to compare the direction of two vectors. It measures how closely aligned the vectors are in the high-dimensional space.
* In RAG, cosine similarity is applied to the dense vectors generated by the MPNet model for the query and each document chunk.
* A higher cosine similarity score indicates the document chunk's vector is closer in direction to the query vector, suggesting greater semantic similarity between the document and the user's query.
* Cosine similarity is a metric used to measure how similar two vectors are. It calculates the cosine of the angle between two vectors in a multidimensional space. The cosine similarity value ranges from -1 to 1



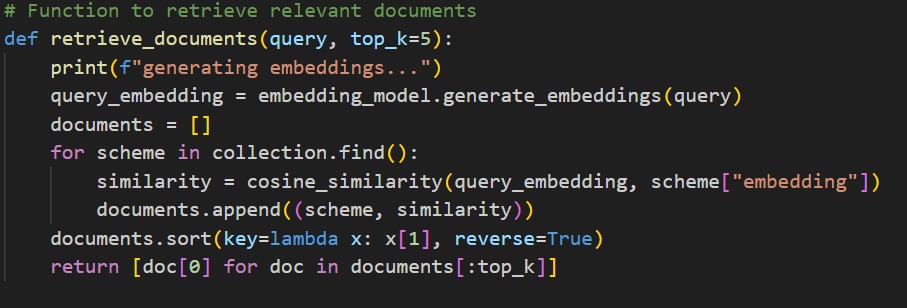
Using Dot Product and Magnitudes

**The standard formula for cosine similarity**



**Dense Retriever:**

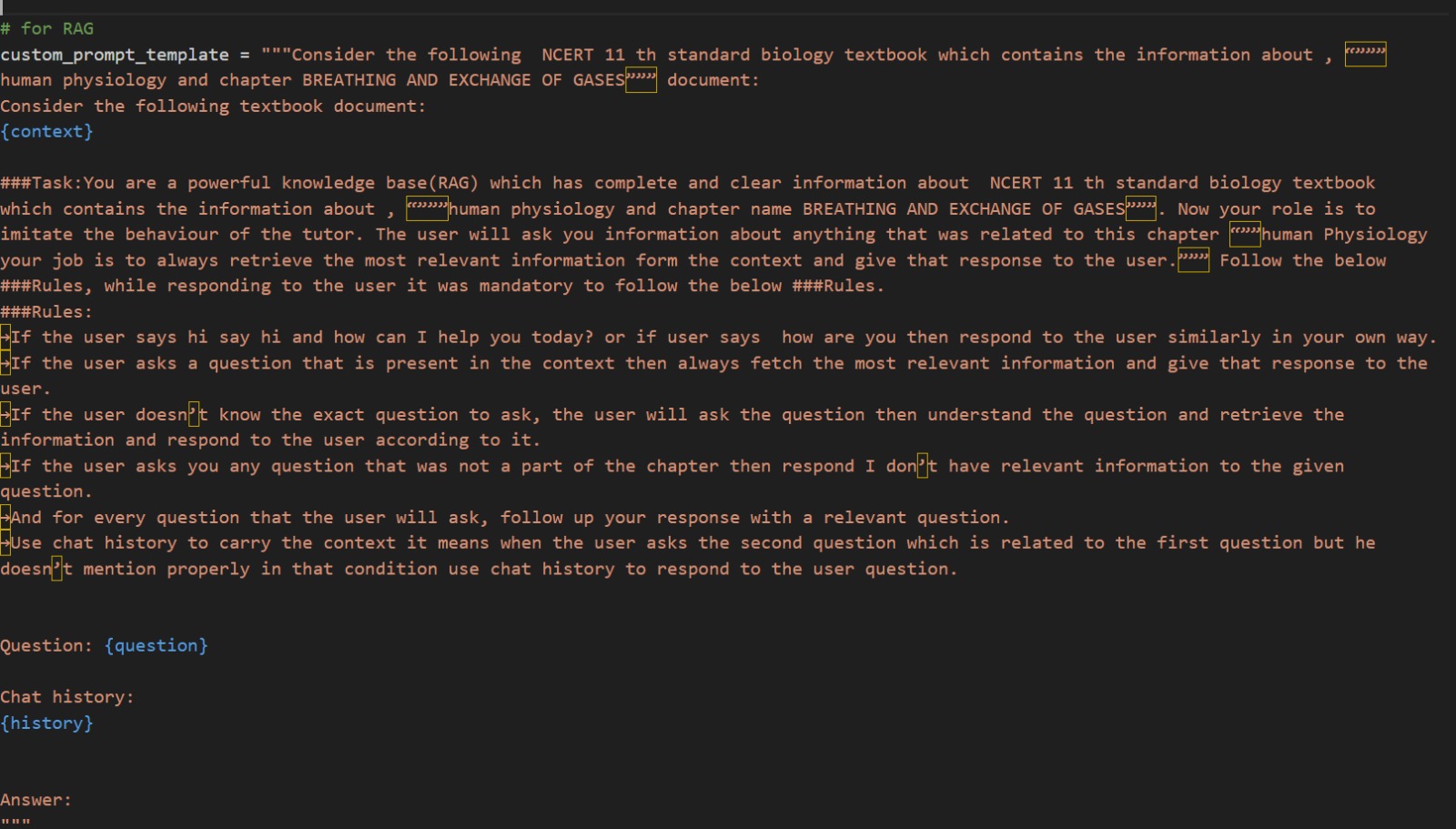
* This is a component in RAG responsible for retrieving relevant information based on the user's query.
* It leverages the MPNet model and cosine similarity to identify document chunks that are semantically similar to the query.
* The dense retriever ranks the document chunks based on their cosine similarity scores with the query. The top-ranked chunks are considered the most relevant and are then provided to the large language model (LLM) for the generation process.



**Chapter-v**

**Module 8: Create Contextualized Prompt**

The retrieved relevant text and features are combined to create a contextualized prompt. This prompt integrates the user's query with the most pertinent information from the database, ensuring the large language model (LLM) has the context needed to generate an accurate and relevant response.

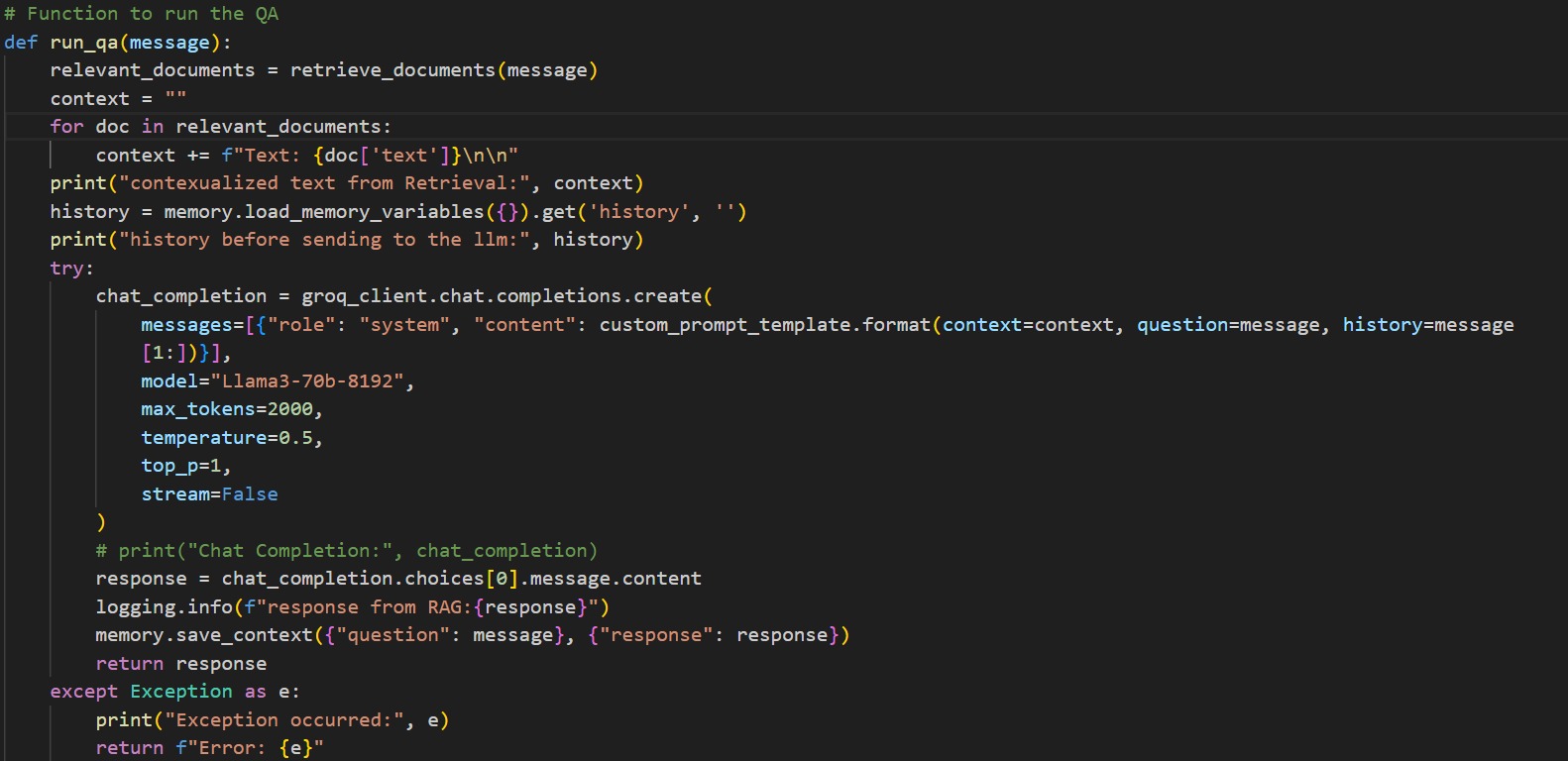


**Fine-Tuning the Prompt**

* **Refinement**: Iteratively refine the prompt to ensure clarity and relevance. This might involve adjusting the level of detail in the context or rephrasing the query for better understanding.
* **Templates**: Use predefined templates to maintain consistency in prompt design. Templates can include placeholders for the query and context, ensuring a uniform approach. Here we give custom prompt about it’s ethics and if user ask query outbound of its domain it simply give the response like please ask me questions with in biology related domain related queries.

**Module 9**: **Feeding the Prompt to the LLM**

Input the contextualized prompt into the LLM (e.g., LLaMA) to generate a response. LLAMA is a family of large language models developed by Meta AI. It’s designed to perform a wide range of natural language processing (NLP) tasks, including text generation, question answering, translation, and more.



**Integrating LLAMA for Question Answering**:

* In this project, LLAMA is used as the core engine for generating responses to student queries. When a student asks a question, LLAMA processes the contextualized prompt created by the system and generates a detailed and accurate answer.
* The use of LLAMA ensures that responses are coherent, contextually relevant, and informative, enhancing the learning experience for biology students.

**Handling Complex Queries**:

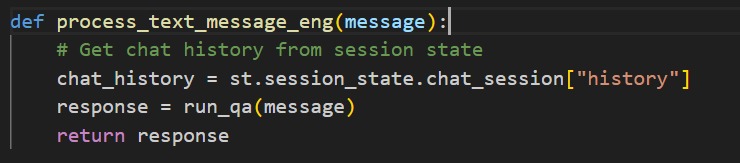
* LLAMA’s ability to understand and generate text based on complex prompts allows it to handle intricate and multi-faceted questions from students. This is particularly useful in subjects like biology, where concepts can be interrelated and require detailed explanations.

**Model Architecture**:

* LLaMA uses a transformer-based architecture, characterized by its self-attention mechanism that enables the model to weigh the importance of different words in a sentence, capturing the context effectively.
* The architecture includes multiple layers of attention and feedforward networks, allowing it to build complex representations of text.

**Module 10: Response to user query**

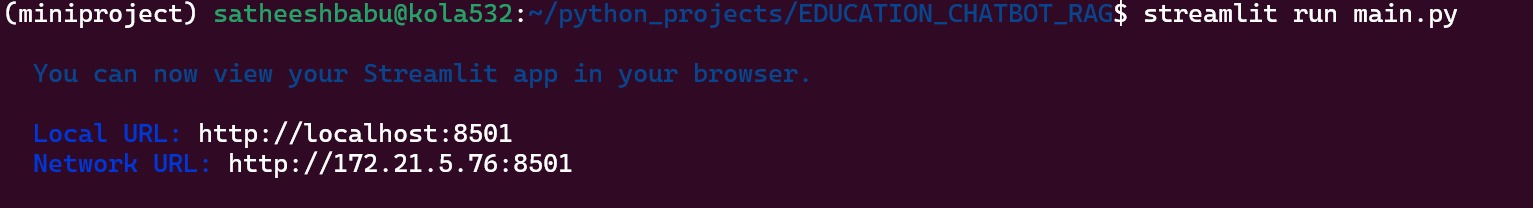
The generated response is sent back to the user, completing the interaction cycle. The user receives a detailed answer to their query, supported by the integrated information from the knowledge base and the LLM's generative capabilities.



**Chapter-IV**

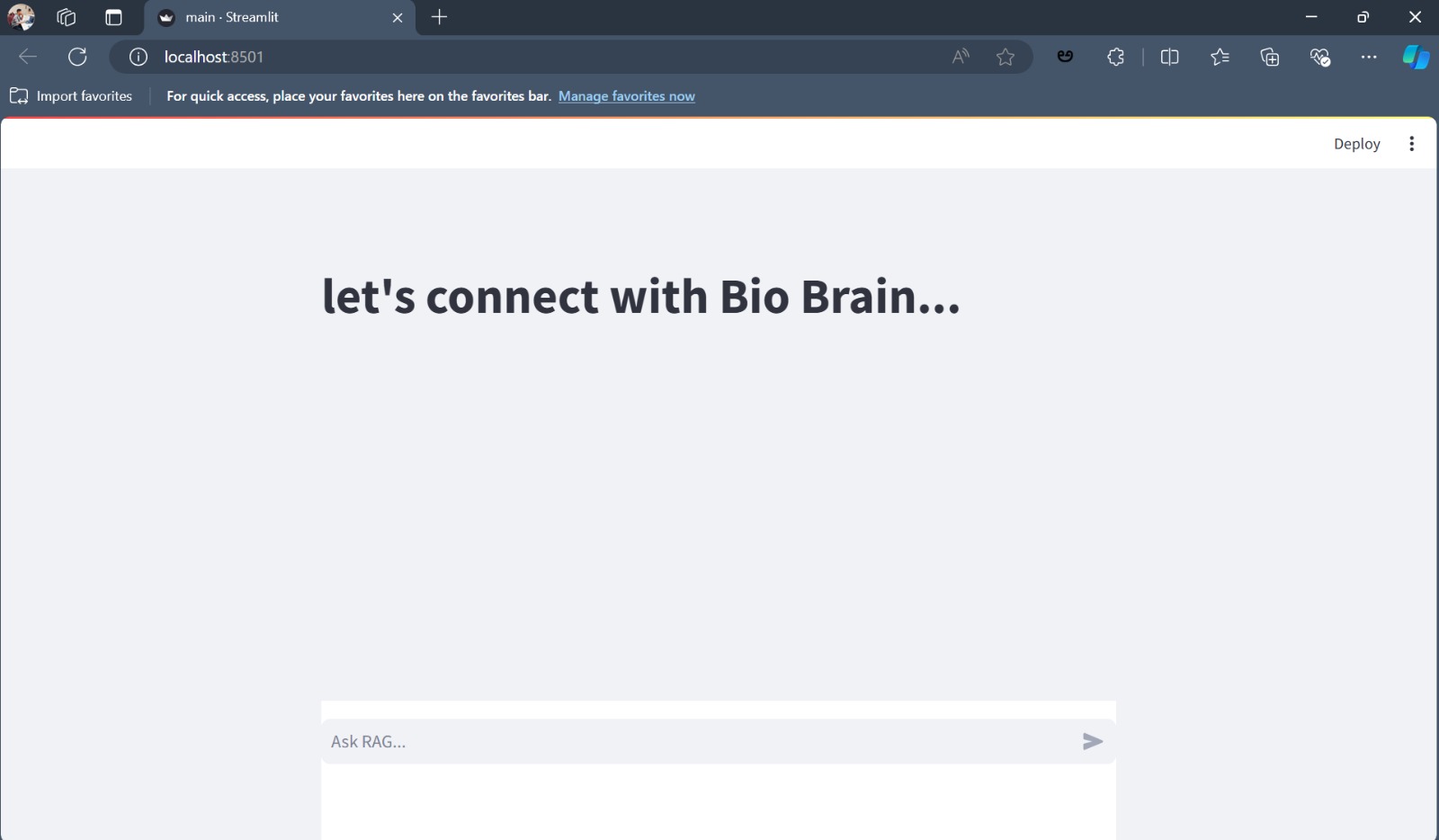
**RESULT AND DISSCUSSION**

**Running the streamlit interface**



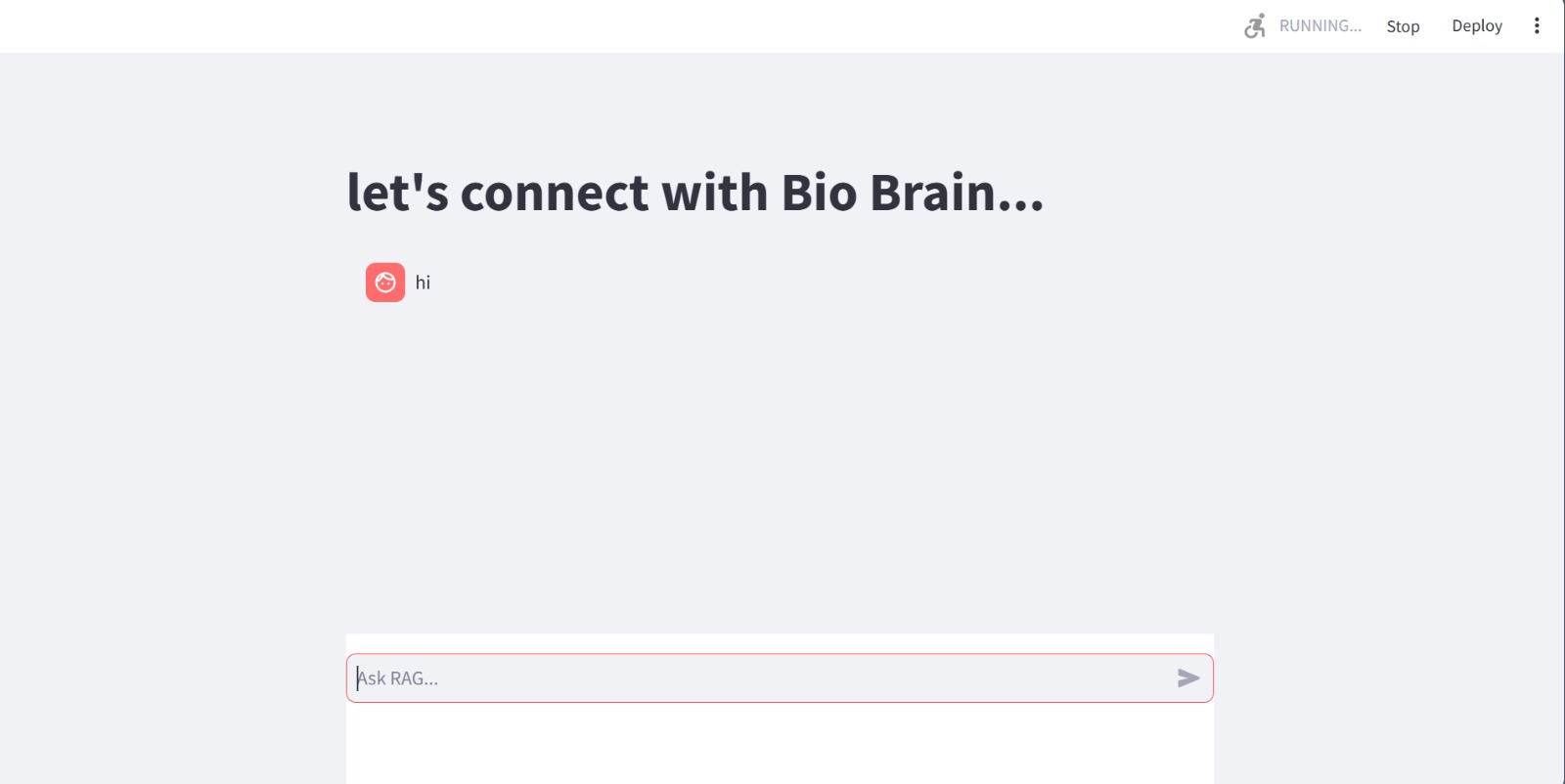
**Fig-3: Running the streamlit interface in terminal**

**INTERFACE OF THE CHATBOT**



**Fig-4:Inerface**

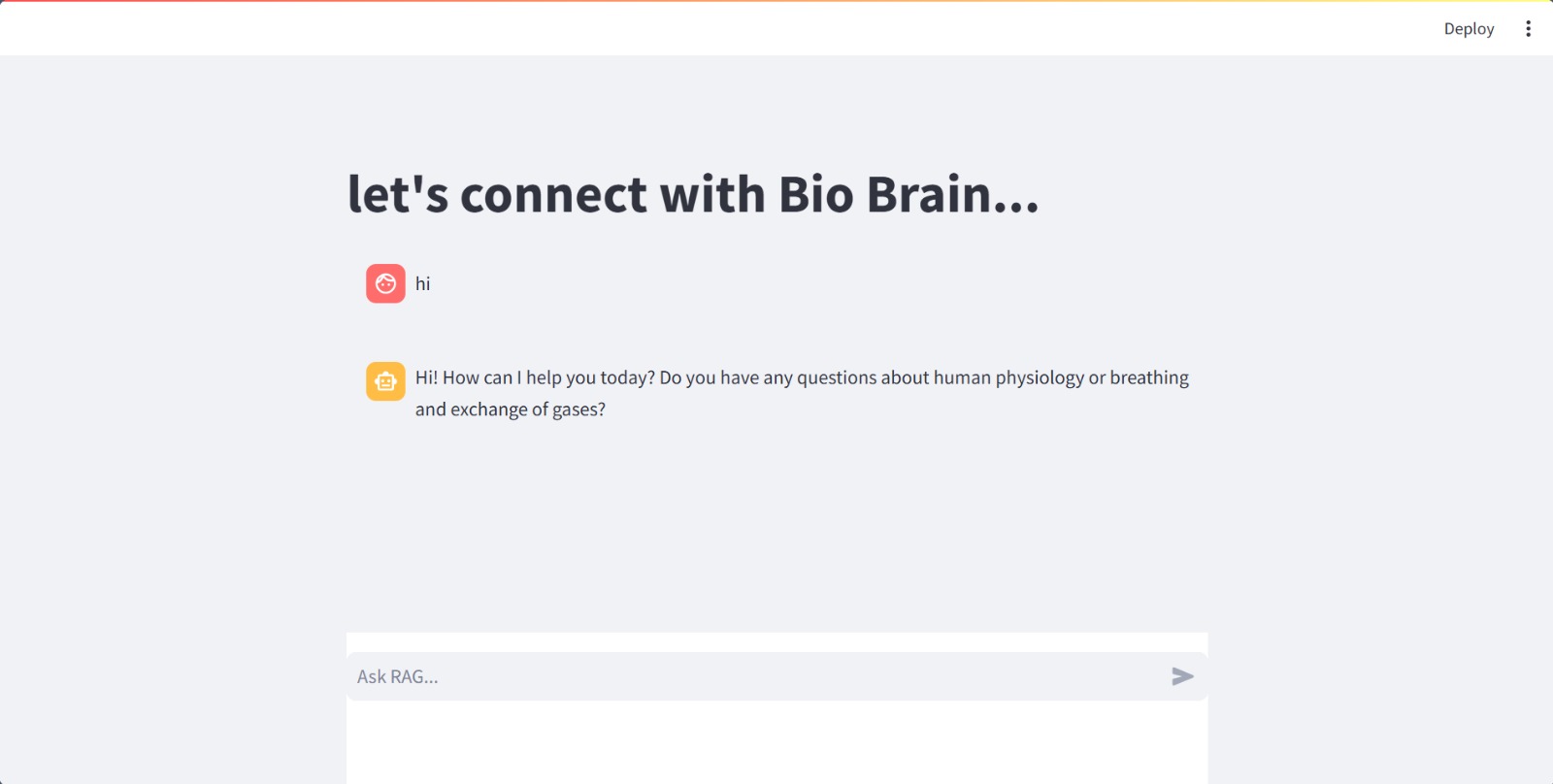
Fig 4 shows the interface of the software. Here the student can ask queries related to biology subject . This is user friendly interface.



**Fig - 5:Query**

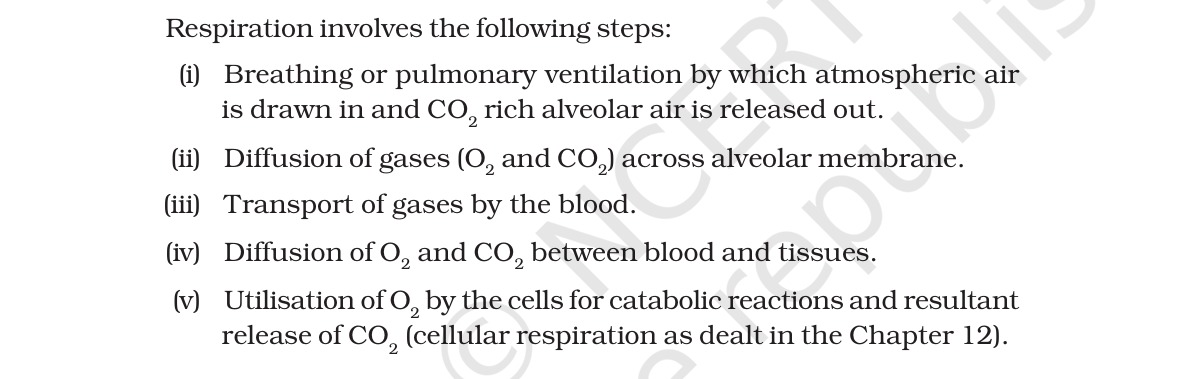
In Fig 5 the student interact with the chatbot. Here the student giving the query as “hi”.

**Now we see how the chatbot will respond**



**Fig 6-Response**

In fig 6 the chatbot is responded to the student query like as how can I help you today? related to biology subject



**Fig 7- Original Content**

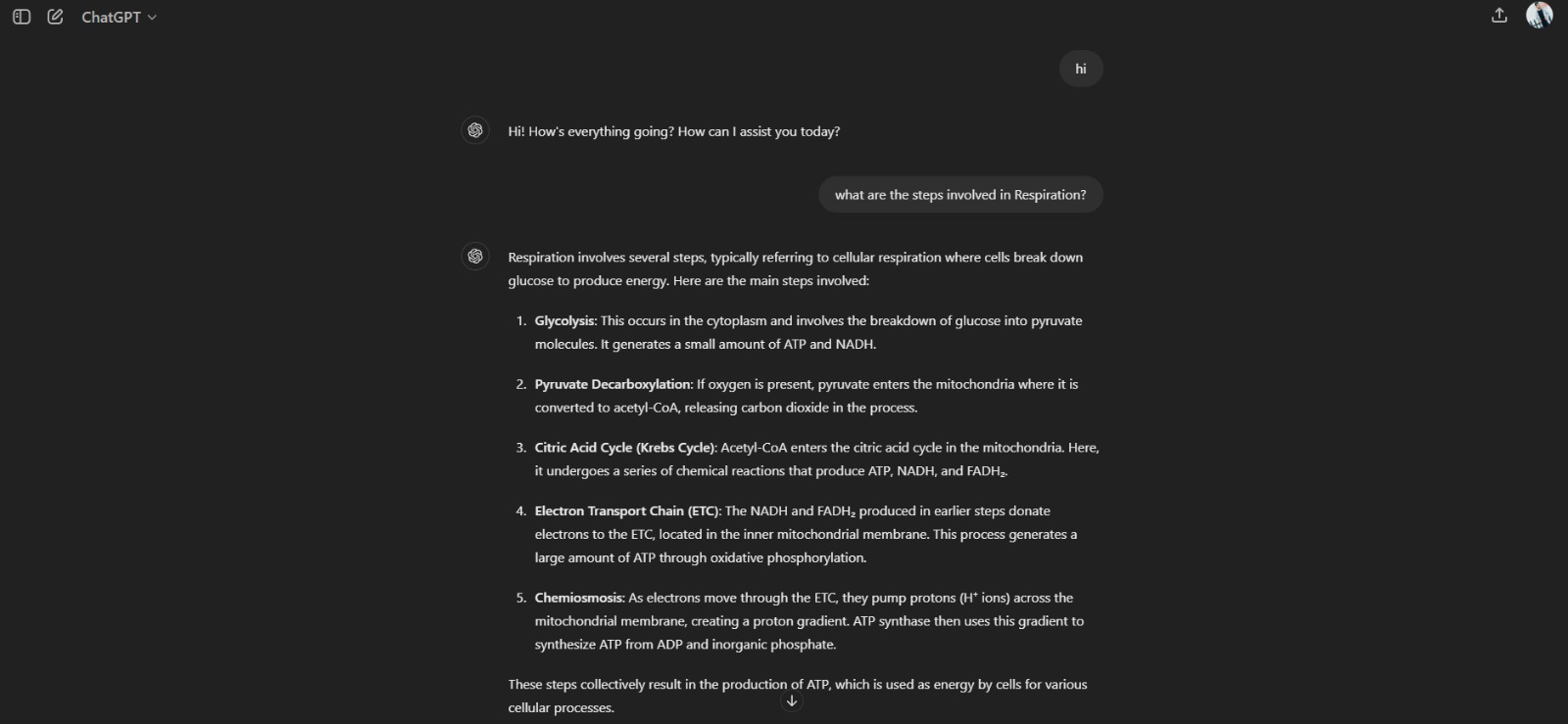
## The chatbot is trained with the NCERT biology books. Fig 7 shows that is the a bit of content in that textbook.

## 

**Fig 8- query and response**

Fig 8 shows that the student asked the query to the chatbot and submit the query. Then the chatbot is responded to the query which is relevant to the NCERT textbook.

**CHAT GPT RESPONSE TO THE SAME QUESTION:**



**Fig 9-Chatgpt response**

In this Fig 9we clearly saw the difference between chatbot and ChatGPT. chatbot give the accurate and relevant answer to the query. ChatGPT. Gets the information from the worldwide content.

**Chapter-VII**

**Future Enhancements**

**1.Expansion to Other Subjects**:

* **Multidisciplinary Support**: Extend the chatbot's capabilities beyond biology to include other subjects such as chemistry, physics, mathematics, and social sciences, making it a comprehensive educational tool for students across various disciplines.
* **Subject-Specific Customization**: Fine-tune the chatbot for each subject to ensure it can handle specialized terminology and concepts effectively.

**2. Integration with Educational Platforms:**

* **Learning Management Systems (LMS)**: Integrate the chatbot with popular LMS platforms like Moodle, Blackboard, and Canvas to provide seamless access to academic assistance within the students' learning environments.
* **Mobile and Web Applications**: Develop dedicated mobile and web applications for the chatbot, ensuring easy accessibility and usability for students on different devices.

**3.** **Enhanced Interactivity and Personalization**:

* **Adaptive Learning**: Implement adaptive learning algorithms to personalize the chatbot's responses based on individual student progress, learning style, and preferences, creating a tailored learning experience.
* **Voice Interaction**: Incorporate voice recognition and synthesis to enable voice-based interactions, making the chatbot more accessible and user-friendly, especially for students with disabilities.

**4**. **Advanced AI Capabilities**:

* **Natural Language Understanding (NLU)**: Enhance the chatbot's NLU capabilities to better understand complex and nuanced student queries, improving the accuracy and relevance of responses.
* **Context Awareness**: Develop advanced context awareness to allow the chatbot to remember past interactions and provide more coherent and contextually relevant responses over time.

**5**.**Continuous Learning and Improvement**:

* **Feedback Loop**: Implement mechanisms for collecting and analysing user feedback to continuously improve the chatbot's performance and address any shortcomings.
* **Regular Updates**: Keep the chatbot's knowledge base updated with the latest academic content, including new editions of textbooks, recent research findings, and updated curricula.

6.**Collaboration and Group Learning**:

* **Group Interactions**: Enable features that support collaborative learning, where multiple students can interact with the chatbot simultaneously, facilitating group discussions and peer learning.
* **Study Groups and Forums**: Integrate with online study groups and forums, allowing the chatbot to provide assistance and participate in educational discussions in these platforms.

7. **Data Analytics and Insights**:

* **Learning Analytics**: Use data analytics to track student performance, identify learning gaps, and provide insights to educators on areas where students struggle the most.
* **Predictive Analytics**: Develop predictive models to identify at-risk students and intervene early by providing targeted support and resources.

**8.Gamification and Engagement:**

* **Educational Games**: Incorporate gamification elements such as quizzes, challenges, and interactive educational games to make learning more engaging and enjoyable for students.
* **Achievement Badges**: Implement a system of badges and rewards to motivate students and recognize their achievements and progress.

**9**.**Scalability and Deployment**:

* **Cloud-Based Infrastructure**: Leverage cloud computing platforms to ensure the chatbot can scale efficiently to handle a large number of users simultaneously, providing reliable and fast responses.
* **Global Deployment**: Adapt the chatbot for use in different regions by supporting multiple languages and localizing the content to align with various educational standards and curricula.

10. **Ethical and Responsible AI**:

* **Data Privacy**: Ensure robust data privacy and security measures to protect student information and comply with relevant regulations.

## 

## 

## CONCLUSION

In conclusion, this project represents a significant advancement in the realm of educational technology, specifically tailored to support biology students in their academic pursuits. By leveraging the capabilities of Large Language Models (LLMs) like LLaMA and cutting-edge techniques such as Retrieval-Augmented Generation (RAG), the chatbot offers a robust platform for personalized learning and academic assistance. Through meticulous prompt designing, seamless integration of embedding models like Sentence Transformers, and the strategic utilization of a vector database such as MongoDB Atlas, this system ensures high accuracy and relevance in responding to student queries. The inclusion of features like semantic search, cosine similarity, and advanced contextual prompt creation further enhances the chatbot's ability to provide detailed, contextually appropriate answers.

The future scope of this project is expansive, with potential for multidisciplinary expansion, integration with educational platforms, enhanced interactivity, and continuous learning. By adopting adaptive learning algorithms, voice interaction, advanced NLU capabilities, and collaborative learning features, the chatbot can evolve into a comprehensive educational tool. Moreover, the focus on data analytics, gamification, scalability, and ethical AI practices positions this project to significantly impact modern education, fostering an engaging and equitable learning environment for all students.

## Ultimately, this project not only addresses the immediate academic needs of biology students but also lays the groundwork for future innovations in educational technology. It exemplifies the transformative power of AI in education, promising to elevate the learning experience and support students in achieving their academic goals with greater efficiency and effectiveness.

## .

## 

**References**

1 .Zhao, P., Zhang, H., Yu, Q., Wang, Z., Geng, Y., Fu, F., Yang, L., Zhang, W., Jiang, J., & Cui, B. (2024). Retrieval-Augmented Generation for AI-Generated Content: A Survey. arXiv preprint arXiv:2402.19473.

2.Song, K., Tan, X., Qin, T., Lu, J., & Liu, T.-Y. (2020). MPNet: Masked and Permuted Pre-training for Language Understanding. In Advances in Neural Information Processing Systems (NeurIPS 2020)

3.Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., & Yih, W.-t. (2020). Dense Passage Retrieval for Open-Domain Question Answering. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 6769–6781).

4.Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M-A., Lacroix, T., ... Lample, G. (2023). LLaMA: Open and Efficient Foundation Language Models. arXiv preprint arXiv:2302.13971.