Build a ChatGPT-type application that teaches students how to use HTML、CSS、JavaScript to create web pages and use multiple PDF files as a knowledge base.

Student name: Shihan ZENG

Candidate number: 267643

Your degree courses: Computer Science and Artificial Intelligence

Department: University of Sussex, School of Engineering, and Informatics

The name of your project supervisor: Professor Martin White

Calendar year of submission: 2024

Statement of Originality

This report is submitted as part of the requirements for the degree of Computer Science and Artificial Intelligence at the University of Sussex. Except where indicated in the text, this report is the result of my own work. This report may be freely reproduced and distributed, provided that the source is acknowledged.

Acknowledgements

Thank you to my supervisor, Prof Martin, for supervising my thesis.

Abstracts

Artificial intelligence has always been a tool of great potential and great popularity. Moreover, artificial intelligence has always been a tool with great potential and popularity.

In this project, a ChatGPT-type application was created to teach students how to create web pages using HTML, CSS, and JavaScript and using multiple PDF files as a knowledge base. A comprehensive knowledge base of vector data was constructed by analysing and understanding the PDF documents. The system also enhances learning by extracting meaning from complex queries and retrieving precise information from the knowledge base through vector databases and the semantic search function of LLM. Initial applications have shown that this LLM-based approach dramatically increases students' interest in learning and speed of skill acquisition while enhancing autonomy and flexibility in learning. The project demonstrates the ability of LLM to understand and generate natural language, proving the great potential of artificial intelligence and machine learning technologies applied to modern education. (Zhi Zhang,2023) By integrating advanced technologies, the project provides personalised learning paths, improves learning efficiency and transforms modern educational resources into a more interactive and efficient format.

Contents

Statement of Originality ***ii***

Acknowledgements ***ii***

Abstracts ***iii***

1.Introduction ***1***

1.1 Background and Importance of the Project: **1**

1.2 Scope: Limitations and Boundaries of the Project **1**

1.3 Problem statement: **2**

1.4 Project Objectives and Scope **2**

1.41 Primary Objectives 2

1.42 Secondary Objectives 1

1.5 Professional considerations **1**

1.51 Professional Considerations: 1

1.52 Ethical Considerations: 1

2.Literature Review ***2***

2.1 Artificial Intelligence in Education **2**

2.11 Personalised Learning Assistant 2

2.12 The Future of Personalised Learning 3

2.13 Interactive Tutoring Systems 3

2.14 Personalisation in Education 4

2.15 Challenges and Future Directions 5

2.16 Legal and Ethical Implications 5

2.2 Integrating AI in Web Development Education **5**

2.21 Evolution and Collaboration in AI-driven Education 5

2.22 The Indispensable Role of Teachers 6

2.23 Synergy between AI and Human Educators 6

2.3 Benefits and Challenges of using PDF as a Knowledge Base **7**

2.31 Benefits of Core Features of Vector Databases 7

2.32 Challenges 7

3 Requirements ***9***

3.1 Requirements Gathering **9**

3.2 Technical feasibility analysis: **9**

3.3 Requirements Analysis **10**

3.4 Requirements Specification **10**

4. Design and Methodology ***12***

4.1 Methodology 12
4.1.1 Technology stack selection 12
4.1.2 User Interface Design (UI/UX) 14
4.2 Design 14
4.2.1 Structural Diagrams 14
4.2.2 Behavioural Diagrams 16
4.3 Architectural Design 18
4.4 Data Modelling 22
4.41 Adoption of Qdrant Vector Database 22
4.42 Cloud Service Deployment Strategy 22
4.5 User Interface Design 23
5. Implementation 25
5.1 Module 1: Knowledge Retrieval for Students 25
5.2 Module 2: PDF Uploading and Processing 27
Implementation Approach 27
Algorithms 27
Data structure 28
6. Testing 36
7. Results and Evaluation 36
8. Conclusions 37
9. Reference 37
10. Appendices 40

1.Introduction

The following section will introduce the project’ s background, importance, limitations, and boundaries. Moreover, the problem statement of the project will be introduced, including the objectives, research scope, and research significance.

1.1 Background and Importance of the Project:

Web skills are fundamental to modern schooling, and since the internet is so widely used, building nearly every web page requires an understanding of HTML, CSS, and JavaScript. In addition to teaching students how to design web pages using web technologies, this project aims to produce an application similar to ChatGPT. It integrates several PDF files as a dynamic knowledge base to provide an interactive, responsive, real-time, and efficient learning platform. AI technology can significantly increase educational efficiency, particularly in personalised learning and information retrieval, according to Chen et al. (2020). According to the study, students may learn complicated ideas more quickly and efficiently by utilising AI to personalise learning paths and information. This project draws on this finding and aims to improve student motivation and efficiency by integrating AI technology into a web-based learning platform to enable fast and accurate knowledge retrieval.

In the past, knowledge took much work to access. However, with the development of technology and technological advances, from the era of no internet to the era of the internet to the current era of AI, the possibilities of accessing knowledge have grown. People can access more and more knowledge to the best of their ability. One of the main reasons is the speed of access; in the era of no Internet, people wanted to get knowledge only through books and the newspaper and so some old methods, and then in the Internet era, people want to get knowledge through the Internet query to get the relevant knowledge, and then to the current era of artificial intelligence, people need what can be delivered directly to the ai on behalf of the labour, or can use them to complete Some problematic tasks, such as writing articles, designing pictures, making plans, etc., knowledge appears in a more accessible and advanced way. In the context of the rapid development of AI, education has also started to change with the emergence of AI in the form of AI, and more often than not, the use of AI in terms of the education sector to facilitate teaching as well as to improve the efficiency of learning has become more and more common. In this context, traditional learning resources such as textbooks face reforms in search of a learning tool that combines traditional education and AI to enhance efficiency. Due to the rapid development of technology, traditional ways of acquiring knowledge have gradually failed to meet people's requirements for the speed of acquiring knowledge because they not only provide limited personalised or interactive learning opportunities but also because the traditional ways of acquiring knowledge are inferior to the combination of AI tools, so with the development of the times, the risk of being eliminated is faced if only the previous traditional education methods are used. For this, there is a need to combine AI and artificial intelligence.

1.2 Scope: Limitations and Boundaries of the Project

This project is focused on providing a foundational understanding of web development for beginners to intermediate learners, emphasising the core technologies of HTML, CSS, and JavaScript. While web development encompasses a wide array of languages, frameworks, and tools, the scope of this tool is intentionally limited to these essential technologies to establish a strong base for further learning. The reliance on PDF documents as the primary content source may restrict the breadth of available information. However, it guarantees the quality and relevance of the material. Additionally, this tool is not intended to replace traditional educational pathways but rather to complement them by offering an alternative method of learning that caters to the digital age.

1.3 Problem statement:

Traditional methods of learning, including textbooks, classroom instruction, and static online resources, present specific challenges that potential hinder learning. Firstly, textbooks can quickly become outdated, leaving learners with gaps in their knowledge and skills (Tony, 2023). Secondly, traditional learning models often lack the interactivity and engagement required to master technical subjects, leading to reduced learner motivation and poor learning outcomes. In addition, one-size-fits-all approaches fail to accommodate individual learning styles and progress, further exacerbating the difficulty of mastering complex coding practices.(Tony, 2023) These challenges highlight the need for innovative educational tools that can adapt to the pace of technological change, engaging learners through interactive content, and providing personalised learning experiences to meet individual needs.

1.4 Project Objectives and Scope

1.41 Primary Objectives

The main goal of this project is to develop an intelligent learning assistant based on ChatGPT, which aims to help beginners and students to acquire new skills and web-related knowledge (HTML, CSS, JavaScript) by providing them with the relevant PDF files or by importing the relevant modules into the knowledge base. During the learning process, users can ask the ChatGPT assistant questions to solve their doubts. In addition, teachers, who are one of the main core user groups, can use the application to teach and integrate web knowledge into the knowledge base, thus improving the learning experience and efficiency of their students.

To achieve the main objectives of the project:

PO1: Develop a ChatGPT-based intelligent learning assistant designed for beginners to help them learn HTML, CSS, and JavaScript by providing them with relevant PDF files or importing relevant modules into the knowledge base.

PO2: Use ChatGPT's intelligent dialogue function to answer real-time questions during the learning process, reducing learning difficulties and improving learning efficiency.

PO3: Provide a platform for viewing and interacting, e.g. expose documents as Google Document, to enhance the user learning experience and interaction with knowledge.

Rationale for achieving the primary objective

Through the above aims we provide an interactive learning environment that makes the learning process efficient and interesting, effectively promoting autonomous learning.

1.42 Secondary Objectives

The project's secondary objectives include supporting educational institutions and teachers by providing additional learning resources and support for students:

SO1: Provide a platform that teachers and educational institutions can use to support and enrich teaching and learning.

SO2: Enhance the technological skills of users (students and teachers) to enable them to make better use of modern technology for learning and teaching.

SO3: To further enhance learning through personalised learning advice and resources based on the user's learning history and problems.

1.5 Professional considerations

1.51 Professional Considerations:

Professional standards were considered and adhered to throughout the completion of the project, ensuring that all activities and decisions align with the highest level of professionalism expected in the field. This includes maintaining transparency, integrity, and accountability in all aspects of the project.

1.52 Ethical Considerations:

Ethical considerations were prioritised by upholding the principles outlined in the British Computer Society (BCS) code of conduct. This entails promoting fairness, honesty, and respect in all interactions and endeavours related to the project. Additionally, measures were taken to safeguard the privacy and confidentiality of sensitive information and data involved in the project.

By integrating both professional and ethical considerations into the project's framework, the project aims to deliver quality outcomes and contribute positively to the broader professional community by upholding values of integrity and responsibility in the field of computer science and technology.

2.Literature Review

Within the module Literature Review, I have divided the thesis into three main modules based on the topic:

1, (Creating a ChatGPT type of application) means that I need to create a chat software like application that uses the LLM language model to answer questions;

2. (Teaching students how to create web pages using HTML, CSS and JavaScript) means that my application will help students learn.

3. (Using multiple pdf files as a knowledge base) means that the answers to the questions are based on the pdf files I uploaded. The knowledge base here uses a vector database:

I will start with the first module of the thesis, describing what ChatGPT's Big Language Model is, and discuss the next modules in turn.

2.1 Artificial Intelligence in Education

The development of educational technology, from Sidney Pressey's mechanical teaching machines in 1923 to sophisticated AI such as ChatGPT, illustrates a relentless pursuit of educational automation, signalling a shift towards a more personalised, accessible, and interactive learning experience. The advent of ChatGPT and similar generative AI technologies represents the latest milestone in this evolution, offering unprecedented opportunities for personalised learning and instruction.

2.11 Personalised Learning Assistant

ChatGPT serves as a generative language model that acts as an all-knowing tutor, personal learning consultant, and intelligent assistant, capable of answering questions, solving problems, and assisting with a range of daily tasks. This versatility facilitates a wide array of learning activities and heralds a significant shift towards the digital transformation of school education (Fang, Wang, & Ma, 2024). By providing personalised feedback, facilitating collaborative learning, and supporting teachers in analysing educational data, AI-powered assistants like ChatGPT undoubtedly enhance the teaching and learning experience.

2.12 The Future of Personalised Learning

The role of GPT models in education extends beyond mere content delivery. They offer an unprecedented opportunity for personalised learning experiences that are adapted to individual student needs, enhancing learning outcomes by tailoring educational content and facilitating interactive educational experiences (Yuan, 2023).

Moreover, Large Language Models (LLMs) such as GPT-3 demonstrate remarkable capabilities in various domains, including education, due to their extensive training on a wide range of internet texts. This training allows them to understand and generate natural language close to that of humans, making them highly effective in simulating conversations, providing information, and assisting in learning processes. The effectiveness of GPT-3 in education can be seen through its applications in personalised tutoring, content generation, and providing intelligent feedback mechanisms. Despite the challenges associated with the integration of GPT models in educational systems, such as lack of interpretability and privacy concerns, solutions like the development of explainable AI (XAI) and robust data protection measures are being proposed. Furthermore, GPT models continue to evolve (Jafarzade, 2023); for example, GPT-4 showcases improved capabilities and raising discussions on their role in educational settings and beyond.

2.13 Interactive Tutoring Systems

Interactive Tutoring Systems (ITS) have shown considerable effectiveness in enhancing the educational experience across various subjects, particularly in computer science education, where they have been found to be more effective than both teacher-led classroom instruction and non-ITS computer-based instruction, indicating a significant overall advantage of ITS (Nesbit et al., 2014). Notably, ITS has almost matched the effectiveness of human tutoring, a significant achievement considering the traditional reliance on human instruction for personalised learning experiences (VanLehn, 2011).

Moreover, the use of ITS in Higher Education (HE) has demonstrated a moderate positive effect on academic learning, outperforming other instruction methods and activities, including traditional classroom instruction and computer-assisted instruction (Steenbergen-Hu & Cooper, 2014). Similar findings are observed in the effectiveness of interactive e-learning environments, which have resulted in better performance and higher satisfaction levels among students compared to traditional and less interactive e-learning methods (Zhang, 2005). Furthermore, several studies highlight the importance of interactive computer tutors in online tools for teaching programming, emphasising the need for designs that educators and developers can use to improve overall teaching effectiveness (Shen, Wohn, & Lee, 2020).

2.14 Personalisation in Education

The integration of AI into the educational sphere marks a pivotal evolution in how learning is personalised and delivered. AI's capacity to tailor educational content according to the unique needs, preferences, and pace of each learner is transforming the educational landscape. AI-driven systems can analyse individual learning patterns and performance, allowing for the optimisation of learning outcomes through customised content delivery. These advancements enhance e-learning modules and pave the way for AI-powered virtual tutors, simultaneously tackling ethical challenges linked to their deployment (Khan, Omar, & Jian, 2023).

AI-based personalised e-learning systems are designed to offer learning content and assessments that are aligned with a learner's comprehension level and preferred modes of learning. Despite the benefits, these systems encounter challenges such as ensuring user privacy, mitigating biases inherent in data and algorithms, and sustaining learning motivation. Addressing these challenges necessitates a comprehensive understanding of personalised education, the effective utilisation of AI's benefits, and ongoing research to navigate future directions (Murtaza et al., 2022).

Furthermore, embracing AI in education can craft learning environments adapted to meet the diverse needs of students, thereby fostering creativity, engagement, and enhanced learning outcomes. Nevertheless, the successful incorporation of AI into educational systems demands a careful consideration of privacy, security, and ethical issues to ensure its responsible use and to guarantee equitable access to quality education for every learner.

Deploying ChatGPT and Large Language Models (LLMs) in educational settings brings forth numerous challenges and considerations that must be addressed to ensure their effective and ethical use. While specific research directly addressing these challenges in education is limited, insights can be drawn from broader discussions on deploying AI and LLMs across various domains. These insights show that the successful integration of ChatGPT and LLM into an education platform is a significant endeavour, however there is still a need to address technical challenges, ensure data privacy, and establish a robust regulatory framework. In addition, pedagogical aspects must be considered during deployment to create a learning environment that is inclusive, equitable, and conducive to an engaging educational experience.

2.15 Challenges and Future Directions

Despite the potential benefits, the integration of AI in education is not without challenges. Issues such as ensuring data privacy, mitigating biases, and addressing the digital divide must be carefully managed to ensure equitable access to AI-enhanced education. Moreover, as AI continues to evolve, there is a need for ongoing research to explore its full implications for the educational environment and to maximise its benefits for learners worldwide(Jiao, 2023). Furthermore, ChatGPT facilitates the digital transformation of school education.

2.16 Legal and Ethical Implications

The legal and ethical implications of employing ChatGPT and similar technologies in education are complex. Questions around the copyright of AI-generated content and the ownership rights over such content highlight the need for clear legal guidelines. Ensuring that these technologies are used ethically, protecting student data privacy, and preventing potential misuse are critical considerations. Furthermore, educators and policymakers must navigate these challenges thoughtfully to harness the benefits of AI in education while mitigating risks.

2.2 Integrating AI in Web Development Education

The following section will elaborate on the topic of teaching students how to create web pages using HTML, CSS, and JavaScript through the use of AI.

2.21 Evolution and Collaboration in AI-driven Education

Zhu (2021) provides a visionary perspective on how AI is poised to transform the educational landscape, particularly highlighting the shift in the teacher's role from primarily disseminating information to becoming facilitators of a more enriched, interactive learning experience. This evolution is propelled by AI's unique capabilities to personalise learning, offering each student a tailored educational journey that is both efficient and effective. AI technologies, through adaptive learning algorithms, can meticulously analyse a student's progress, strengths, and areas needing improvement, adjusting the curriculum in real-time to match their learning curve.

Moreover, AI's ability to automate routine tasks such as grading, attendance, and even certain aspects of classroom management enables teachers to devote more time to fostering critical thinking, creativity, and interpersonal skills among students. In addition, the potential for AI to support psychological counselling and provide insights into students' learning habits and preferences opens new avenues for holistic education that addresses both academic and emotional well-being.

2.22 The Indispensable Role of Teachers

Concurrently, as Jumankuziev Uktamjon (2023) argue, the essence of teaching, especially in disciplines as dynamic and practical as programming languages, cannot be fully replicated by AI. Teachers bring an irreplaceable depth of knowledge, not just in technical subjects but in pedagogy, critical thinking, and real-world applications of theoretical concepts. Their expertise in selecting programming languages and frameworks that are not only academically relevant but also in high demand within the job market. Teachers' understanding of industry trends, coupled with their ability to impart practical experience, shapes students into not just coders, but problem solvers, innovators, and thinkers. This human touch enables the cultivation of a learning environment that encourages questions, fosters innovation, and stimulates intellectual curiosity beyond what AI currently offers.

2.23 Synergy between AI and Human Educators

The future of education, especially in the fast-evolving field of web development, lies in a synergistic approach that leverages the strengths of both AI and human educators. While AI can provide a personalised, efficient, and data-driven foundation for learning, teachers can enrich this learning with context, experience, and the human insight necessary for deep understanding and innovation.

Incorporating AI into the teaching of web development not only introduces students to the technical skills needed to navigate the digital world but also prepares them for a future where technology and human creativity intersect in unprecedented ways. By working hand-in-hand, AI and educators can create a dynamic, responsive, and deeply engaging educational experience that equips students with the skills, mindset, and adaptability required for success in the digital age.

In conclusion, the interplay between AI-driven platforms and traditional teaching methodologies presents a compelling paradigm for the future of education in web development and beyond. It underscores the importance of balancing personalised, technology-enhanced learning with the irreplaceable value of human insight, experience, and mentorship. This balanced approach can optimise educational outcomes and ensure that students are well-equipped for the complexities and opportunities of the modern technological landscape.

2.3 Benefits and Challenges of using PDF as a Knowledge Base

This section outlines the use of multiple PDFs as a knowledge base, whereby the core features and challenges of vector databases are outlined drawing upon Pan, Wang, and Li (2023)'s research.

2.31 Benefits of Core Features of Vector Databases

First, vector databases effectively compress the size of the original data by converting data into vector form for storage, while retaining the key characteristics. This is especially important for text-rich PDF files, which often contain large amounts of instructional materials and programming guides. Storage in vector form reduces the amount of storage space required and provides the basis for fast retrieval.

Second, vector databases can quickly process the query to locate the most similar query data items. This is particularly important in educational technology applications because it allows learners instant access to the information they need, whether it is specific programming concepts, code samples, or solutions.

2.32 Challenges

Nevertheless, they still face the following challenges.

1. efficient data storage: Vector databases effectively compress the size of the original data by converting the data into vector form for storage, while retaining the key characteristics of the data. This is especially important for text-rich PDF files, which often contain large amounts of instructional materials and programming guides. Storage in vector form reduces the amount of storage space required and provides the basis for fast retrieval. The terms "function" and "method" are used interchangeably in some programming languages, however they may be considered as different entities in a vector database. This ambiguity requires databases to be able to recognise the terms and to understand their meaning in a particular context to ensure that the search results

accurately reflect the learner's query intent. Addressing this challenge requires databases to have more advanced natural language processing capabilities to recognise and understand the semantic links between different terms.

2. fast query processing ability: using the mathematical properties in the vector space, vector databases can quickly process the query to find the most similar query data items. This is particularly important in educational technology applications because it allows learners instant access to the information they need, whether it is specific programming concepts, code samples, or solutions.

3. Large Vector Scale: Converting educational resources into vector form improves retrieval efficiency and accuracy, but each vector may contain thousands or even tens of thousands of dimensions, resulting in unusually large vector scales. Managing these large vectors not only challenges storage space, but also increases the complexity and resource requirements for computing similarities during queries. In addition, programming language learning resources cover a wide range of topics and concepts, making natural partitioning and effective indexing difficult. The lack of efficient indexing mechanisms increases retrieval time and affects user experience. Optimising vector representations with efficient similarity algorithms, along with more refined classification and indexing strategies, or automatically partitioning data using machine learning techniques, can improve retrieval speed and accuracy

4. Semantic similarity ambiguity: To show the most relevant resources, vector databases must compute the similarity between the query vector and each vector in the database. This computational process extremely becomes resource-intensive and time-consuming when dealing with large-scale datasets, especially in educational technology applications that require real-time responses. Using techniques such as the Approximate Neighbor Search (ANN) algorithm can reduce computational costs while maintaining the search result's accuracy.

5. High Cost of Similarity Comparison: A learner's query may contain the programming language used and vector data requirements. Such hybrid queries require the database to be flexible enough to handle multiple data types and to understand the compounding intent of the query. Developing more sophisticated query parsing and execution strategies can effectively handle such queries while maintaining accuracy.

6. Lack of Natural Partitions for Indexing: Query processing capabilities and the ability to support large-scale datasets are the foundational pillars that enable the effective management and retrieval of a large number of educational resources encapsulated in PDF format. Despite the benefits of using it, these technological advances have been accompanied by significant challenges.

It also suffers from drawbacks, namely: ambiguity of semantic similarity, management of large vector sizes, high costs associated with similarity comparisons, indexing difficulties due to the lack of natural partitioning, and the complexity of answering "hybrid" queries, which are huge obstacles that require innovative solutions.

3 Requirements

With the popularity of artificial intelligence, the field of education is also advancing with the times, in order to keep up with the pace of the times, people are more and more inclined to choose personalised AI-assisted teaching and learning tools, but in order to investigate people's needs, a questionnaire survey is used here to find out the needs of online beginners.

3.1 Requirements Gathering

The collected needs were analysed through the opinions of online beginners, teachers and students. It is noteworthy that 80 per cent preferred the use of AI in education, while only 20 per cent preferred traditional educational methods. Based on this preference, learning aids incorporating AI will be developed for use by 80 per cent of teachers, students and beginners, as well as resources in the original language for the 20 per cent who prefer to read traditional books.

3.2 Technical feasibility analysis:

The goal of this project is to facilitate teachers, students and web beginners who are learning web knowledge (HTML, CSS, JavaScript). Teachers can upload the knowledge they teach to the Knowledge Base and students can ask questions through the Knowledge Base. Web beginners can also upload PDF files to plan one or more PDF pages to enhance their understanding of the learning process. Here natural language techniques are mainly used to segment the PDFs and then obtain the text and embed it in a vector database for subsequent semantic search and answer generation.

3.3 Requirements Analysis

Next, the information collected was then analysed in order to create an intelligent personalised learning assistant that combines with Ai in order to implement a real-time corresponding Ai intelligence to help these web beginners to solve problems or answer questions. According to their needs, the application should be easy to use, user-friendly and clear, and able to find answers quickly. For this purpose, if the teacher wants to create a learning plan for the students and place the learning materials in the knowledge base to be provided to the students, the system should be an application that is able to search for the materials quickly and respond in real time to provide answers. To implement this technology, the application involves Natural Language Processing (NLP) techniques to segment the material, analyse it and embed the text in a vector database for semantic search. This is followed by sorting and matching the relevant answers and finally by Large Language Modelling (LLM) to generate answers to an application. Use these for the purpose of implementing this application.

3.4 Requirements Specification

With a needs analysis, we refine these needs to fulfil our goals better. The table below details this web page's functional feasibility and non-functional requirements.

Functional feasibility

ID	Requirement	Description	Priority
FR1	Knowledge Base Upload	Teachers can upload educational content to the knowledge base.	high
FR2	Query-based learning	Students can ask questions and get answers from the teacher's uploaded knowledge base.	high
FR3	PDF uploading and processing	Web beginners can upload PDFs to enhance their learning. The system segments the PDF, performs a	high

		semantic search, and generates answers.	
FR4	Learning Module Selection	Learning aids with integrated AI for 80 per cent of users and traditional text content for 20 per cent of users.	high
FR5	User Interface	Provide an intuitive and user-friendly interface for uploading content, querying the knowledge base and navigating the learning modules.	high

Non-functional requirement

ID	demand (economics)	descriptions	priority
NFR1	usability	This site provides answers to user questions.	height
NFR2	flexibility	Ability to scale to accommodate an ever-increasing number of users, content and queries.	height
NFR3	performances	Ability to process user requirements quickly.	height
NFR4	user friendliness	Ability to access and use the site in different sizes.	height

4. Design and Methodology

This section summarizes the project's design, including the architecture, data model, user interface design, and other aspects. The project effectively integrates various technologies and design strategies to build a cohesive system, providing a simple, efficient, and interactive learning experience. The design consists of the following five parts:

Methodology: Discusses the selection of the technology stack, including LangChain, Qdrant, and Streamlit, and how they integrate into the application.

Design Diagrams: Includes UML class and component diagrams, providing an overview of the system's structure and relationships between components.

Architecture Design: Explains the interactions between internal components, and how they handle front-end input, back-end data retrieval, and more.

Data Model: Covers the strategies for storing, processing, and retrieving text data, particularly the use of the Qdrant database and its cloud deployment strategy.

User Interface Design: Emphasizes the simplicity and responsiveness of the UI design to ensure users can easily navigate and use the system.

4.1 Methodology

4.1.1 Technology stack selection

This application involves the knowledge of LLM, which uses LangChain, Qdrant, and Streamlit techniques to integrate this ChatGPT application.

LangChain is an open-source framework designed for developing applications that rely on the LLM. It can be used to build and deploy applications based on large language models that handle language comprehension and generation tasks. Such language models are formed by pre-training on large datasets to form complex deep learning models capable of responding to user queries, such as answering questions or generating images based on textual cues. Furthermore, LangChain provides a range of tools and abstraction mechanisms designed to enhance the customisation, accuracy, and relevance of the information generated. Developers can use LangChain's components to create new prompt chains or customise existing templates. In addition, LangChain includes tools that enable large language models to access new datasets without additional training. In simple terms LangChain is about combining large amounts of data, and its main purpose is to be used to retrieve information relevant to a query in a vector store and to allow LLMs to be easily referenced with as little computational effort as possible. For example, it works by taking a large data source, such as a 50-page PDF file, breaking it up into chunks, and embedding them into a Vector Store. How to create a prompt-completion, when there is a large document vectorised representation, he will be able to work with the LLM, once Langchain retrieved the relevant information in the Vector Store, we use it and prompts together with the feed to the LLM, to generate our answers, so this framework is very important for this LLM application.

Qdrant is an open-source high-performance, highly scalable vector database with fast vector search and efficient data storage. It is mainly used to store and manage large-scale high-dimensional vector data, such as images, text, and audio. It adopts a storage structure based on vector indexes, which allows fast retrieval of similar vector data and supports real-time insertion and query operations. It also provides a rich set of APIs and tools for data import, queries, and management. Qdrant aims to provide users with a high-performance, easy-to-use, and open vector database solution to help them quickly build applications and services based on vector data. For Langchain, Qdrant can help provide more effective support for searching and querying data in the language chain, thus improving user experience and system performance; it also provides easy-to-use APIs and client tools that can be quickly deployed and used. In this project Qdrant can provide back-end support for LangChain, especially when there is a need to efficiently retrieve and manage enormous amounts of vector data. For example, Qdrant's vector search capabilities are especially important if LangChain handles applications that need to quickly retrieve documents or entries that are most relevant to a user's query.

When langchain and Qdrant are used in combination, LangChain manages and processes the logic and output of language models, while Qdrant is responsible for efficiently storing and retrieving the vector data generated or processed by these models. This combination can provide a powerful and efficient backend solution for developing complex AI applications.

Teachers upload to the knowledge base through Qdrant, and our project hosts the integrated Qdrant on top of a cloud service. This cloud-based hosting solution features a manageable cluster of Qdrant databases for users. This SaaS solution allows teachers to easily deploy and manage Qdrant databases in the cloud without worrying about infrastructure maintenance and management.

Streamlit is an open-source Python library for quickly building user interfaces for data science and machine learning applications. It helps developers quickly build interactive applications without having to write a lot of UI code. Key features of Streamlit include an easy-to-use syntax, real-time updates, and automatic layout. It also provides rich data visualisation and interactive components to easily create dashboards and applications. The most important thing is that Streamlit can be used in conjunction with machine learning models such as LLM, integrating them into the application to achieve the role of LLM and Streamlit integrating.

4.1.2 User Interface Design (UI/UX)

The main target users of this project are beginners in web development, so my UI design here is as simple and clear as possible, easy to navigate the design, so that the interface is clear, can make the user quickly and intuitively find the functions and information they need, here the use of streamlit multi-page design, in the sidebar can easily click to open the different pages and use. The responsive design also ensures that users can use the site smoothly across different pages and devices, and that the use of the site is not affected by device changes. The streamlit deployable website feature is used here so that the URL can be easily accessed on different screen sizes.

4.2 Design

The following section outlines the design by drawing upon class and component diagrams to explain the relationships between the components.

4.2.1 Structural Diagrams

Figure 4.1 describes the components of the system and their attributes, methods, and relationships with each other.

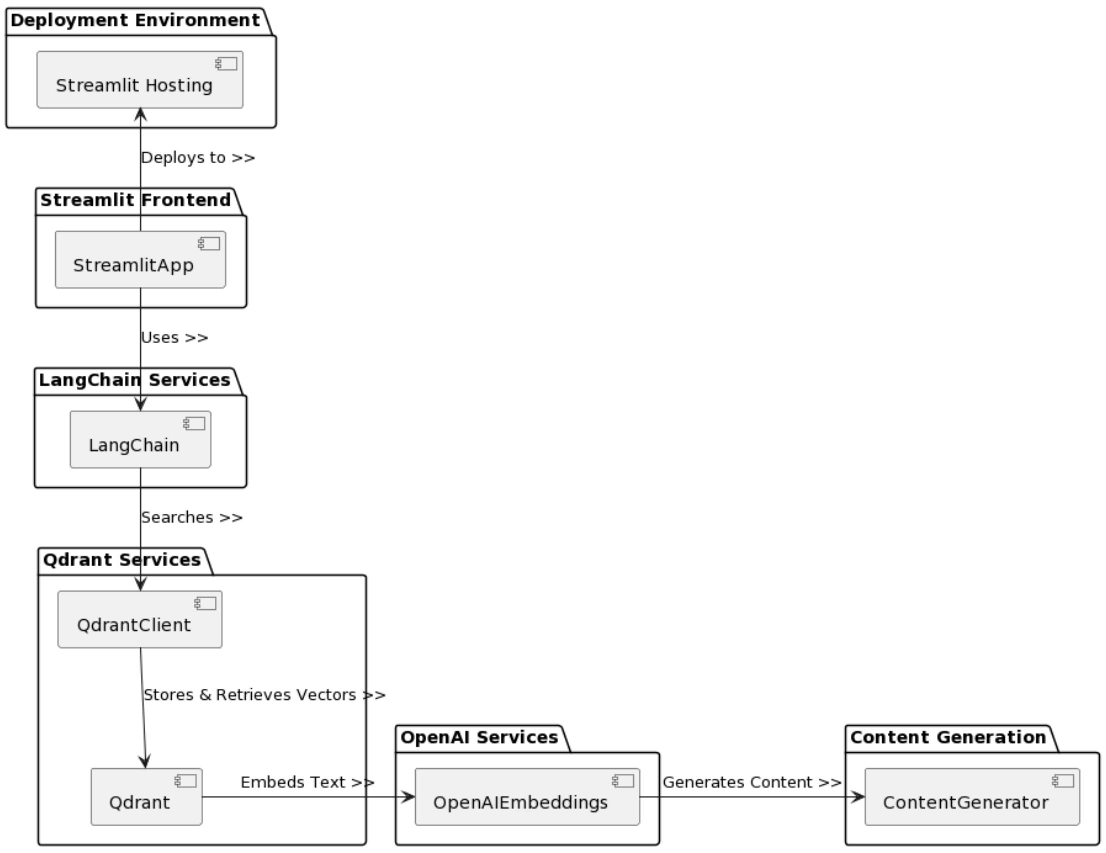


Figure 4.1 Component Diagram

Figure 4.1 illustrates a component diagram to represent the relationship between different components and below is an explanation of the workflow of each part.

Deployment: The application is deployed to the Streamlit hosting platform.

Front-end operations: Users interact with the application through the StreamlitApp, e.g. entering questions, uploading PDFs, etc.

LangChain Processing: LangChain receives input from the front-end and processes it, e.g., generates vector embeddings, performs semantic search.

Qdrant Service: QdrantClient stores and retrieves text vector data from the Qdrant database.

Content Generation: The content generation module generates content based on user input and calls OpenAI services for text embedding and content generation.

4.2.2 Behavioural Diagrams

Two case diagrams show the different functionalities of the web development process. One of the cases outlines the core approach of the thesis, where the teacher imports knowledge into a database and students use the knowledge base to query and learn from it, thus making the learning process more efficient. Below is a description of each component and its interactions in the class diagram: In the use case diagram in Figure 4.2, two main actors are defined: “Student” and “Teacher”, which have different interaction and operation rights in the quiz system.

Submit Query: Students can submit questions to the system. This may involve entering a text query, such as a question about an academic topic.

Import Content to Database: Teachers can upload academic content, such as essays or research materials, to the system database in PDF format. This process involves text processing and vectorisation for subsequent search and retrieval.

Ask Questions: Students can also ask the teacher a question directly. This may be a more specific query that requires a more personal response from the teacher. In addition, the use case diagram includes back-end processes that handle query and answer generation.

Text Processing: The text processing process may involve extracting the content of a PDF document and breaking it down into manageable chunks of text.

Generate Embeddings: Generating embeddings refers to converting blocks of text into a machine-understandable vector form. This is usually done by using deep learning models such as OpenAI.

Perform Semantic Search: Performing a semantic search uses a vector representation of the question to find the most relevant content in the database.

Return Answer: The return-to-answer process involves selecting the most appropriate search results and presenting them in a human-readable form to the student asking the question.

Store Vectors: Storing vectors means storing vector representations of text in the system's vector database for later search and retrieval.

Upload PDF: Students upload documents.

Asking questions: After uploading the PDF, students can ask questions.

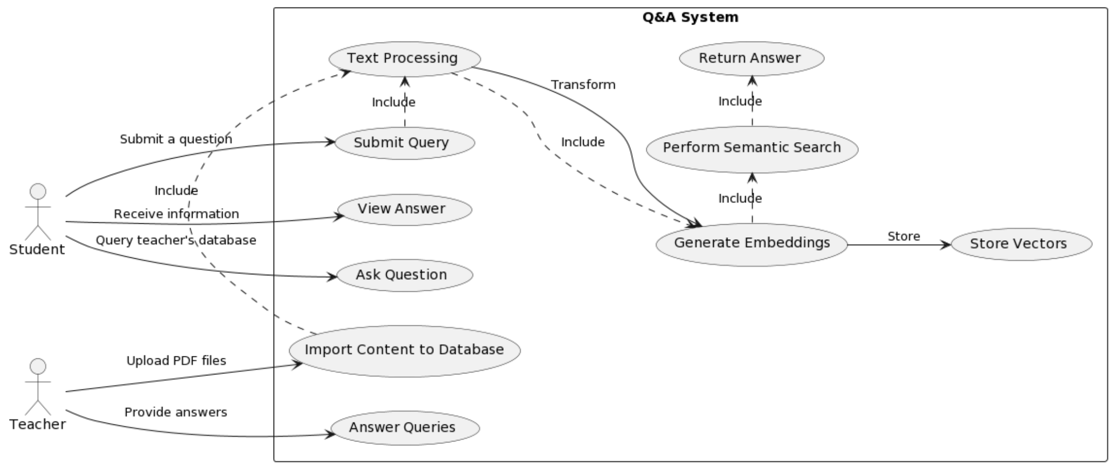


Figure 4.2 Use Case Diagrams:

Figure 4.3 outlines the Use Case Diagram. The sample diagram shows the main actions of a student or web beginner. The processing within the system consists of the following steps:

1. Process PDF as text blocks: The system will process the uploaded PDF into manageable text blocks.

2. Embedding text blocks into the model: the system converts text blocks into vector form for semantic analysis.

3. Storing vectors to Qdrant: the embedded processed text blocks are stored in the Qdrant database.

4. Generating question embeddings: the system converts the questions posed by the students into embedding vectors.

5. Perform a semantic search: the system uses question embedding in the Qdrant database to find relevant blocks of text.

6. Ranked search results: searched text blocks are ranked according to relevance.

7. Generate Answers: The system generates answers from the highest ranked block of text.

8. View Answers: Students view the answers returned by the system.

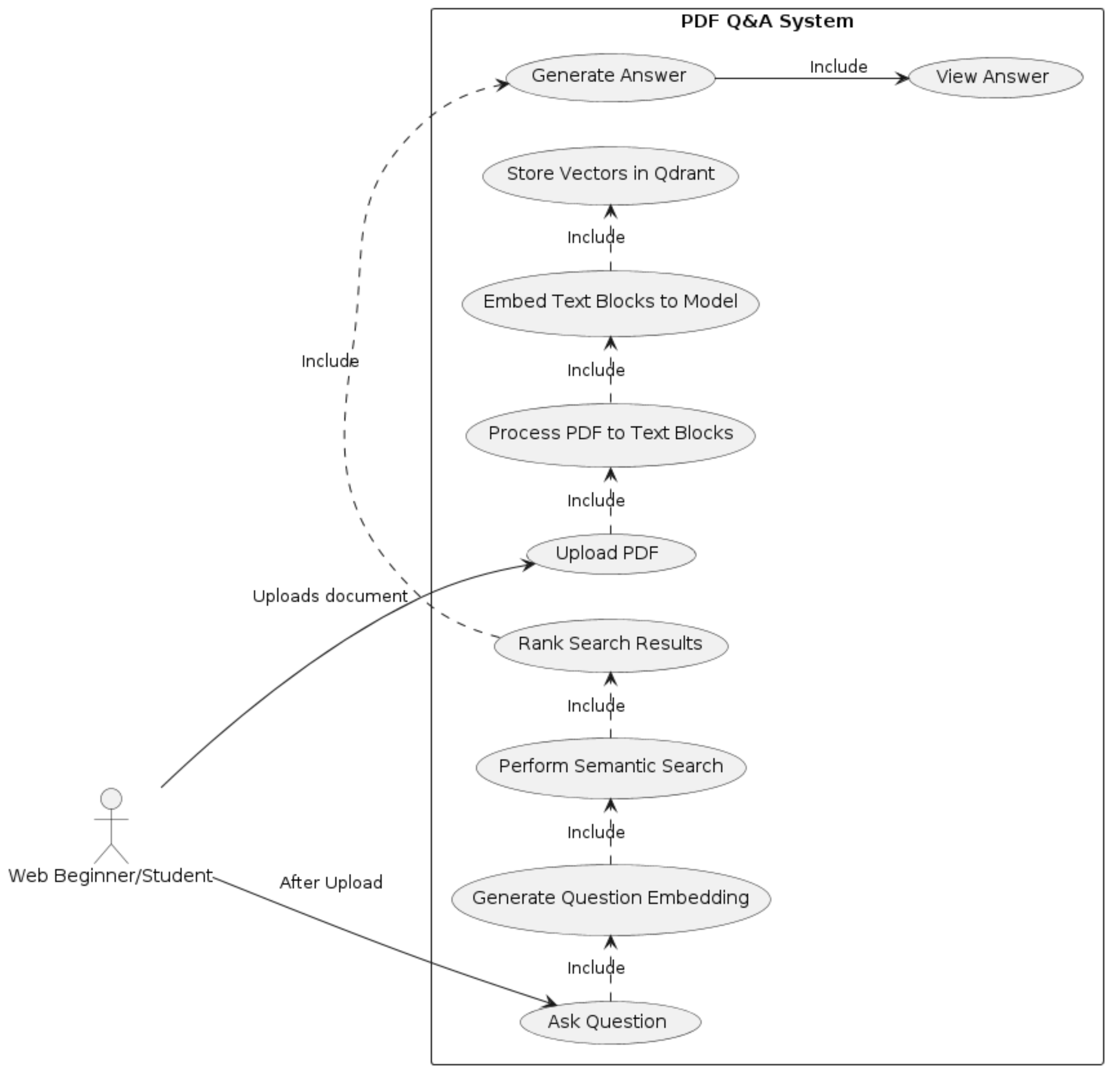


Figure 4.3 Use Case Diagrams

4.3 Architectural Design

In this section, I have used System Context Diagrams, which are diagrams used to describe the relationship between a system and external entities such as users, other systems, and stakeholders.

Here, I have drawn a total of two System Context Diagrams which correspond to two scenarios, respectively.

Figure 4.4 outlines the teacher uploading the knowledge within the database and students asking questions through the knowledge base. Moreover, Figure 4.5 highlights the student/beginner freely uploading a PDF file and asking and answering questions based on the content of the file.

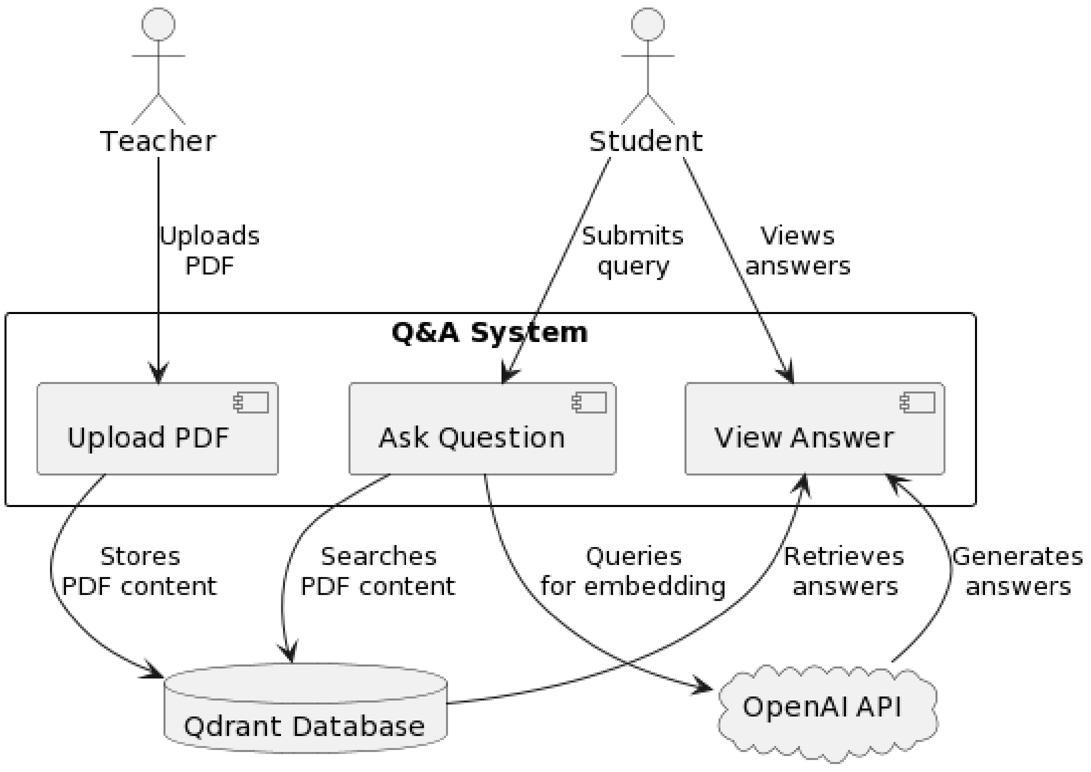


Figure 4.4 System Context Diagrams

Figure 4.4 The steps in interacting with the system are :

1. The teacher uploads PDF files to the system.

2. Students can submit questions and view the answers provided by the system.

3. The system contains a processing upload PDF, question query, and answer to show the function.

4. Qdrant database is used to store PDF content uploaded by teachers and provide search functions.

5. The OpenAI API is used to generate embedding vectors for questions or to generate content for answers.

6. Students retrieve information by submitting queries to the Qdrant database, while the system uses the OpenAI API to aid in answer generation.

Alternatively, students are free to upload PDF files and query and generate responses, as shown in Figure 4.5.

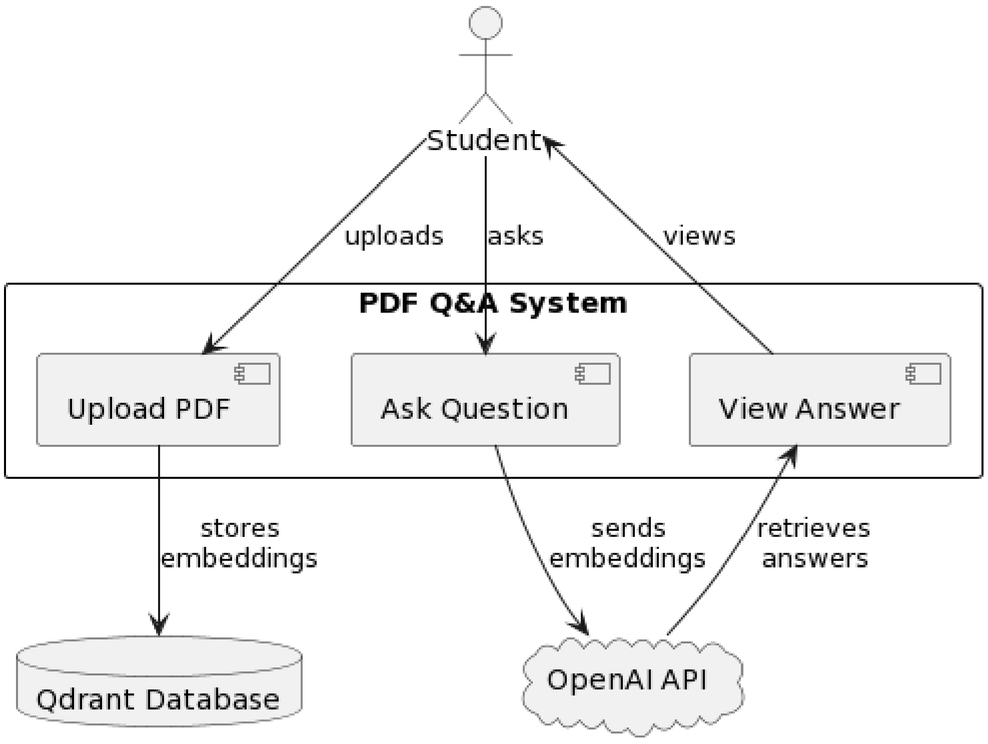


Figure 4.5 System Context Diagrams

This system context diagram represents students' functions:

1. Students upload PDF files, ask questions, and view answers.
2. The PDF Q&A system is the core system responsible for processing PDF documents, generating question embeddings, performing semantic searches and generating answers.
3. The Qdrant database is an external database for storing text embedding vectors, used by the PDF Quiz System.
4. The OpenAI API is an external service used to generate text embeddings and answers.

This diagram shows how students interact with the PDF Quiz System and how the PDF Quiz System interacts with external services.

4.4 Data Modelling

4.41 Adoption of Qdrant Vector Database

In this project, one of our main technical challenges was efficiently storing and retrieving text data to support semantic similarity searches. To address this issue, we chose the Qdrant vector database as our core data storage solution. Qdrant is specifically designed for vector data and provides efficient similarity search capabilities, making it highly suitable for our project needs. By converting processed text blocks into vectors and storing them in the Qdrant database, we not only optimised search efficiency but also enabled rapid similarity matching within large datasets.

4.42 Cloud Service Deployment Strategy

To reduce server load and enhance the scalability of our application, we decided to migrate data storage tasks to the cloud by utilising Qdrant's API. This strategy will facilitate future deployments of the application to platforms like Streamlit, thereby easily accommodating increases in user numbers and data volume. Colab notebooks as a medium for data upload allow us to manually upload files to the Qdrant database, effectively bypassing the complexity of direct vector database operations.

Maturity of the Technology

Although vector databases excel at handling high-dimensional data and supporting fast similarity searches, the relative immaturity of the technology and documentation posed challenges for data restoration and management. To overcome these difficulties, we conducted extensive research and testing to ensure effective use of the database's advanced features.

Information Scarcity in CRUD Operations

Faced with a scarcity of information on CRUD operations for vector databases, we took a proactive approach by exploring and experimenting with API usage methods. Through trial and error in real applications and adjustments, we gradually resolved technical issues and optimised the data operation processes, ensuring the smooth progress of the project.4.5 User Interface Design.

Practical Solution for Data Restoration and Viewing

Qdrant vector database was used to address data storage and retrieval challenges in our project, which shows how we optimised the performance and scalability of the entire application through cloud services and technical solutions.

Given the limitations of vector database technology and the difficulty in directly restoring and viewing large amounts of vector data within the database, we adopted a practical intermediary solution. By using Colab notebooks as a medium for data upload, we were able to manually upload files to the Qdrant database, effectively bypassing the complexity of direct vector database operations.

Colab link: link

4.5 User Interface Design

The main goal of designing a user interface is to provide the user with an intuitive, easy-to-use, and responsive interface to the web application.

Streamlit was used because it has a clean appearance and it benefits from fast development. We have successfully built an aesthetically pleasing and feature-rich user interface to support complex text processing and vector retrieval tasks. In addition, I have used the following design in Streamlit and have shown and explained it in high fidelity diagrams.

I'm creating a multi-page structure here, with page folders: place each page as a separate Python script in a dedicated folder (e.g. pages/), and Streamlit will automatically recognise these scripts as different pages of the application.

Here I am using Streamlit, which has the advantages of fast development and clean appearance, and using these advantages, we have successfully built an aesthetically pleasing and feature-rich user interface to support complex text processing and vector retrieval tasks. streamlit.experimental_get_query_params and streamlit.experimental_set_query_params.

In addition, I have used the following design in Streamlit and have shown and explained it in high fidelity diagrams: Optimize user navigation

Sidebar: Use Streamlit's sidebar (st.sidebar) to provide users with a navigation menu that allows them to easily switch between pages.

Dynamic Titles: Enhance the user's navigation experience by setting page titles and icons on each page using the st.set_page_config method.

The final results are shown in Figures 4.8, 4.9 and 4.10 below.

Homepage

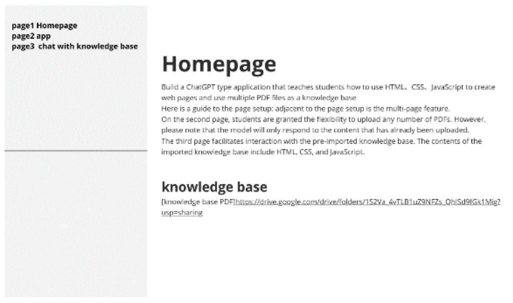
- - **Navigation bar**: Provides a page navigation bar to make it easy for users to switch between different pages and optimise the user experience.
- - **Welcome Text**: Contains a text description that instructs the user on how to use the application, ensuring that the user understands the application functionality.
- - **Knowledge Base Link**: Provides a direct link to the knowledge base, which users can click to get more information.

Chat interface (Chat with multiple PDFs)

- **-Upload area**: designed a clear upload area; users can drag and drop PDF files to the designated area or browse files to upload.
- **-Text input**： In the main part of the window, the user can enter questions in the text box. The design is simple, clear, and easy to navigate.
- **-Button Design**： “Process” Buttons are prominently placed so that when a user uploads a file they can click on it to start processing the document.

Database Query Interface (Ask your remote database)

- **- Query Input**： Provides a simple input box, allowing the user to enter the PDF content of the query.
- **- Layout**： The overall layout is simple, ensuring that users focus on entering questions and receiving answers.


page1 Homepage
page2 app
page3 chat with knowledge base

Homepage

Build a ChatGPT type application that teaches students how to use HTML、CSS、JavaScript to create web pages and use multiple PDF files as a knowledge base
Here is a guide to the page setup: adjacent to the page setup is the multi-page feature.
On the second page, students are granted the flexibility to upload any number of PDFs. However, please note that the model will only respond to the content that has already been uploaded.
The third page facilitates interaction with the pre-imported knowledge base. The contents of the imported knowledge base include HTML, CSS, and JavaScript.

knowledge base

[knowledge base PDF]https://drive.google.com/drive/folders/152Va_4vTLB1uZ9NFZs_QhlSd9lGk1Mig?usp=sharing

Figure 4.1

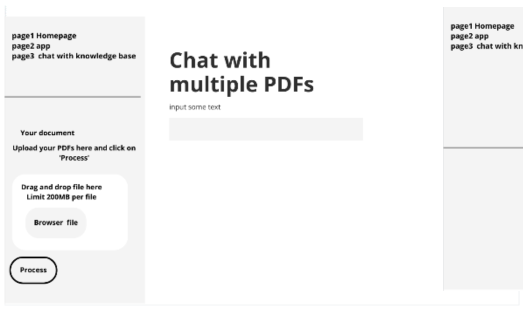


Figure 4.9

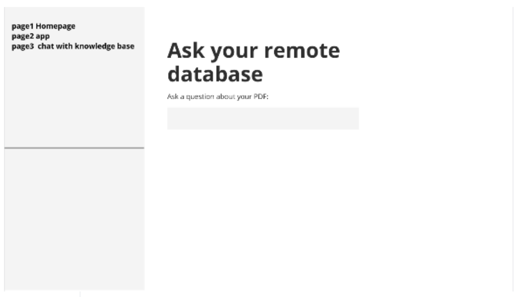

page1 Homepage
page2 app
page3 chat with knowledge base
Ask your remote database
Ask a question about your PDF:

Figure 4.10

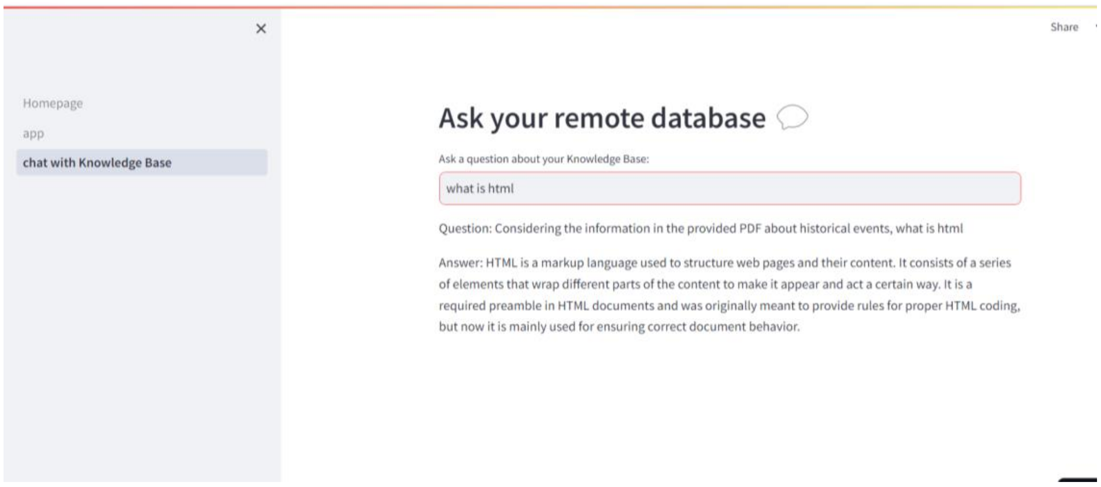
5. Implementation

The implementation phase transformed the conceptual designs and requirements into functional modules.

5.1 Module 1: Knowledge Retrieval for Students

Module 1 is the functionality for interacting with the knowledge base, which makes use of the multi-page functionality in streamlit by placing it in the 'chat with Knowledge Base' page in streamlit, which can be found by navigating through the sidebar, and in which the main implementation is to embed the knowledge base using a large-scale language model, and to use the Qdrant similarity search algorithm that allows us to semantically identify specific support sets or blocks of text that are similar to the user's query. According to Figure 5.1.3 it can be seen here that PyMuPDF, an external library, is used to divide the PDF into many fast pieces, each of which has a size of 512 text and an overlapping block size of 200 text. Using the embedding stored in the support database. The algorithm efficiently searches for relevant information based on user input. This information is then used to generate answers relevant to the user's query using a large language model. Next, screenshots of the user's page and related code snippets are provided. Additionally, the algorithm data structure in Module 1 is the same as the algorithm data structure in Module 2; therefore, this section is primarily discussed in Module 2 before outlining the main user page screenshots and associated code snippets.

The main user page screenshots include: Figure 5.1.1 Home page, Figure 5.1.2 Knowledge base question page, and Figure 5.1.3 Code snippet for uploading a file to the knowledge base.



Ask your remote database

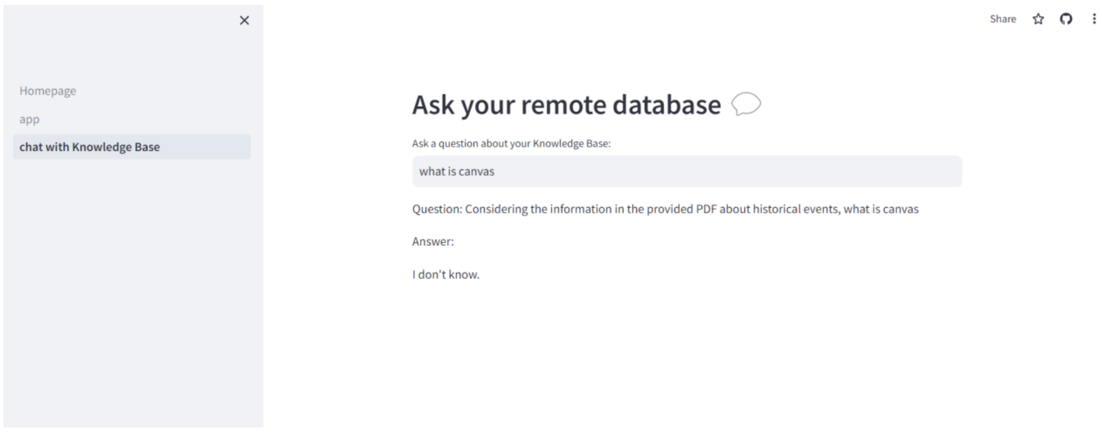
Ask a question about your Knowledge Base:

what is html

Question: Considering the information in the provided PDF about historical events, what is html

Answer: HTML is a markup language used to structure web pages and their content. It consists of a series of elements that wrap different parts of the content to make it appear and act a certain way. It is a required preamble in HTML documents and was originally meant to provide rules for proper HTML coding, but now it is mainly used for ensuring correct document behavior.

Figure 5.1.1 Home Page Interacting with the Knowledge Base



Ask your remote database

Ask a question about your Knowledge Base:

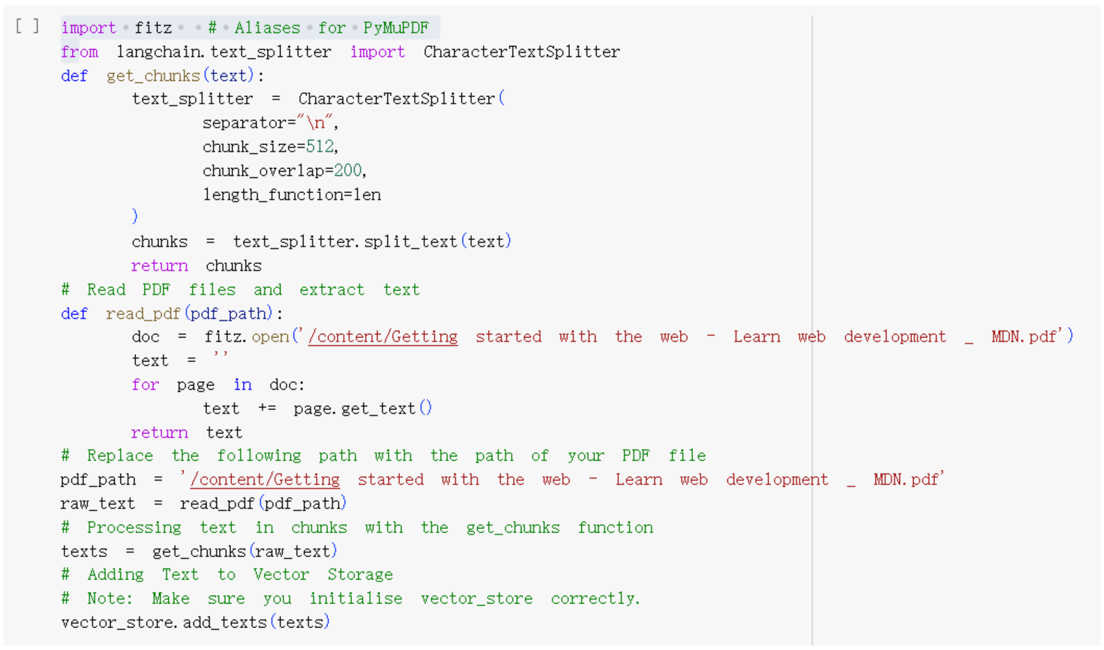
what is canvas

Question: Considering the information in the provided PDF about historical events, what is canvas

Answer:

I don't know.

Figure 5.1.2 Answering questions based on knowledge from the knowledge base



```
[ ] import fitz   # Aliases for PyMuPDF
    from langchain.text_splitter import CharacterTextSplitter
    def get_chunks(text):
            text_splitter = CharacterTextSplitter(
                    separator="\n",
                    chunk_size=512,
                    chunk_overlap=200,
                    length_function=len
            )
            chunks = text_splitter.split_text(text)
            return chunks
    # Read PDF files and extract text
    def read_pdf(pdf_path):
            doc = fitz.open('/content/Getting started with the web - Learn web development _ MDN.pdf')
            text = ''
            for page in doc:
                    text += page.get_text()
            return text
    # Replace the following path with the path of your PDF file
    pdf_path = '/content/Getting started with the web - Learn web development _ MDN.pdf'
    raw_text = read_pdf(pdf_path)
    # Processing text in chunks with the get_chunks function
    texts = get_chunks(raw_text)
    # Adding Text to Vector Storage
    # Note: Make sure you initialise vector_store correctly.
    vector_store.add_texts(texts)
```

Figure 5.1.3 Uploading PDFs into the Knowledge Base

5.2 Module 2: PDF Uploading and Processing

Implementation Approach

We designed a web interface that allows users to upload PDF files and ask questions about their contents. The front-end interface is built using the Streamlit library, and the text retrieval and Q&A functions are implemented on the back-end using OpenAI's language model and the Qdrant vector search engine. This provides an overview of the PDF parsing process, how the text is extracted and split, and the way the system handles queries for new uploads.

Algorithms

PDF Text Extraction: Utilises PdfReader from the PyPDF2 library to iterate through each page of the uploaded PDF documents and extract text. This enables the system to work with the raw content of the documents.

Text Chunking: Employs CharacterTextSplitter to divide the extracted text into manageable chunks. This is crucial for processing large texts that may not fit into the model's context window in one piece.

Prompt Engineering: Leverages PromptTemplate to format the input in a way that guides the language model to provide context-aware responses. The prompt instructs the model to answer questions based on the document text and to cite relevant parts of the text, ensuring concise answers.

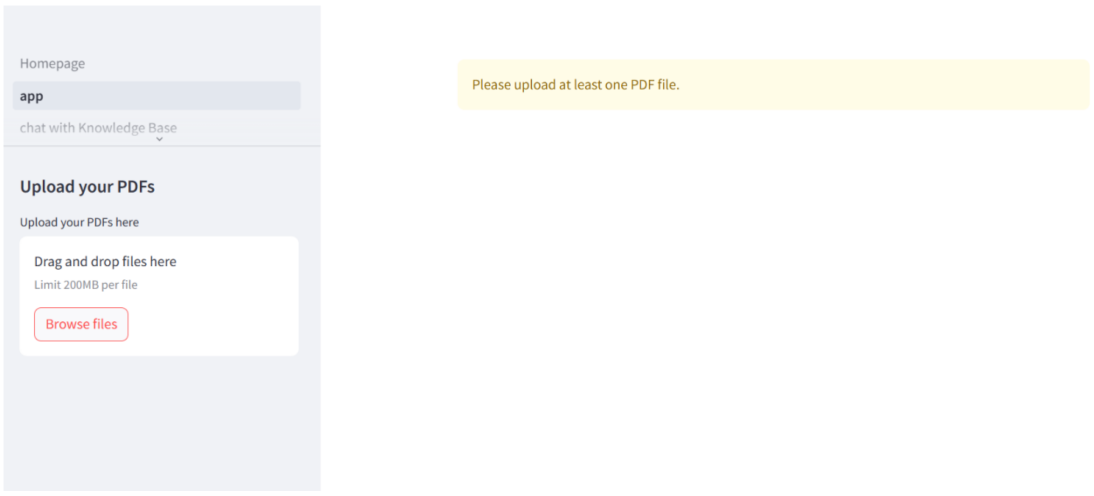
Data structure

Text Representations: After the text is extracted and chunked, it is converted into vector representations using OpenAIEmbeddings. These vectors are used to match the text chunks with user queries effectively.

Conversational State Management: Uses ConversationBufferMemory to store the history of the conversation, maintaining the context across multiple interactions. This is essential for building a cohesive conversation history that the conversational model can refer to.

Screenshots of the interface show the main page (Figure 5.2.1), the upload page (Figure.5.2.2), the text splitting (Figure 5.2.5), and Q&A process (Figure 5.2.7) between applications.

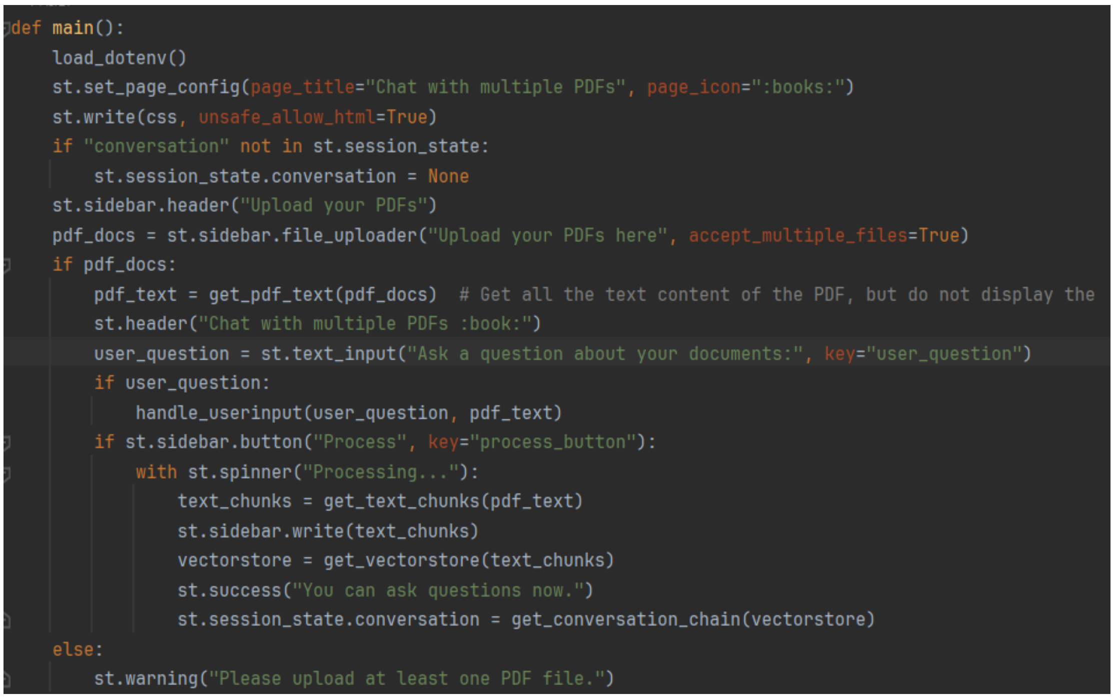
This interactive web application uses Streamlit functions to define user interface elements in Python code, such as text input boxes in Figure 5.2.2 and buttons and file uploaders with yellow warning text in 5.2.1, and a green success message box after successfully uploading a file and clicking on 'process' in Figure 5.2.5. With the green success message box after 'process', this application enables a responsive interactive system where the user can easily upload files, ask questions, and get feedback in response.



Please upload at least one PDF file.

Figure 5.2.1: Main Interface Display

The main logical code snippet of the processing page in Figure 5.2.1 is shown in Figure 5.2.3.



```
def main():
    load_dotenv()
    st.set_page_config(page_title="Chat with multiple PDFs", page_icon=":books:")
    st.write(css, unsafe_allow_html=True)
    if "conversation" not in st.session_state:
        st.session_state.conversation = None
    st.sidebar.header("Upload your PDFs")
    pdf_docs = st.sidebar.file_uploader("Upload your PDFs here", accept_multiple_files=True)
    if pdf_docs:
        pdf_text = get_pdf_text(pdf_docs)  # Get all the text content of the PDF, but do not display the
        st.header("Chat with multiple PDFs :book:")
        user_question = st.text_input("Ask a question about your documents:", key="user_question")
        if user_question:
            handle_userinput(user_question, pdf_text)
        if st.sidebar.button("Process", key="process_button"):
            with st.spinner("Processing..."):
                text_chunks = get_text_chunks(pdf_text)
                st.sidebar.write(text_chunks)
                vectorstore = get_vectorstore(text_chunks)
                st.success("You can ask questions now.")
                st.session_state.conversation = get_conversation_chain(vectorstore)
    else:
        st.warning("Please upload at least one PDF file.")
```

Figure 5.2.3

In Figure 5.2.2 shows the text input box that is displayed when a file has been uploaded.

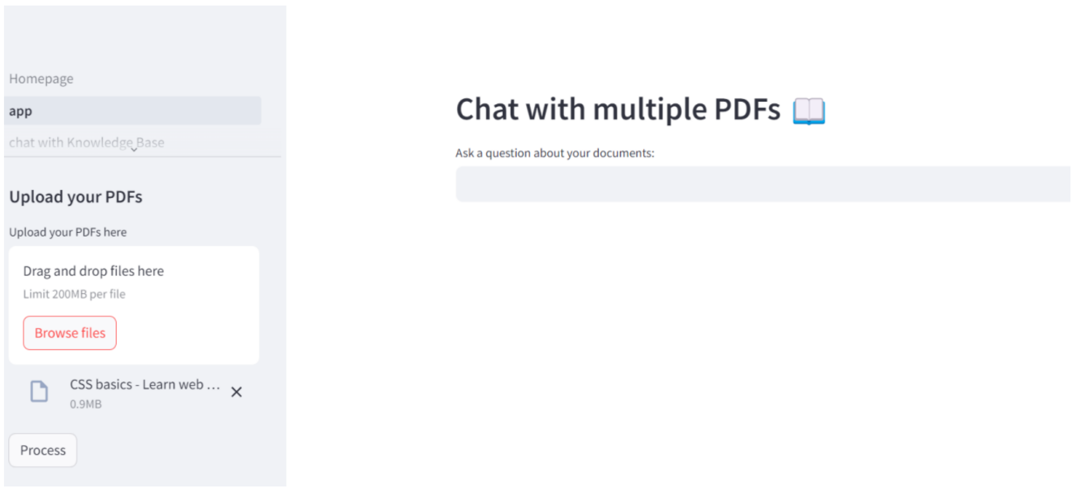


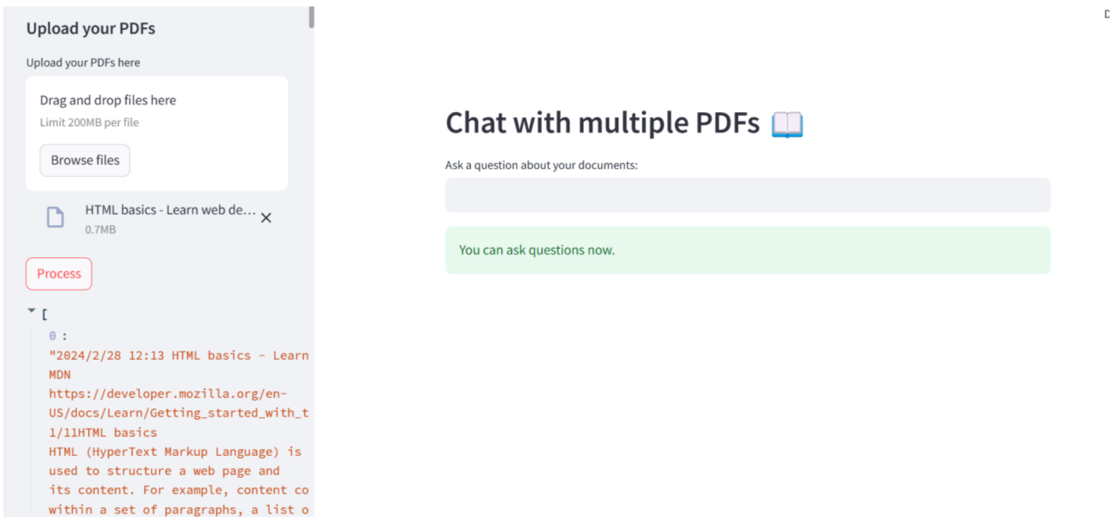
Figure 5.2.2: Uploading files

The key code snippet in Figure 5.2.2 is shown in Figure 5.2.4:

```
user_question = st.text_input("Ask a question about your documents:", key="user_question")
```

Figure 5.2.4

The page in Figure 5.2.5 and the code snippet in Figure 5.2.6 mainly show the logic in terms of file processing, where the uploaded PDF file is broken down into many chunks of about 1000 words each, and they overlap to the extent of 200 words, and finally embedded into a vector database.



Upload your PDFs

Upload your PDFs here

Drag and drop files here
Limit 200MB per file

Browse files

HTML basics - Learn web de...
0.7MB

Process

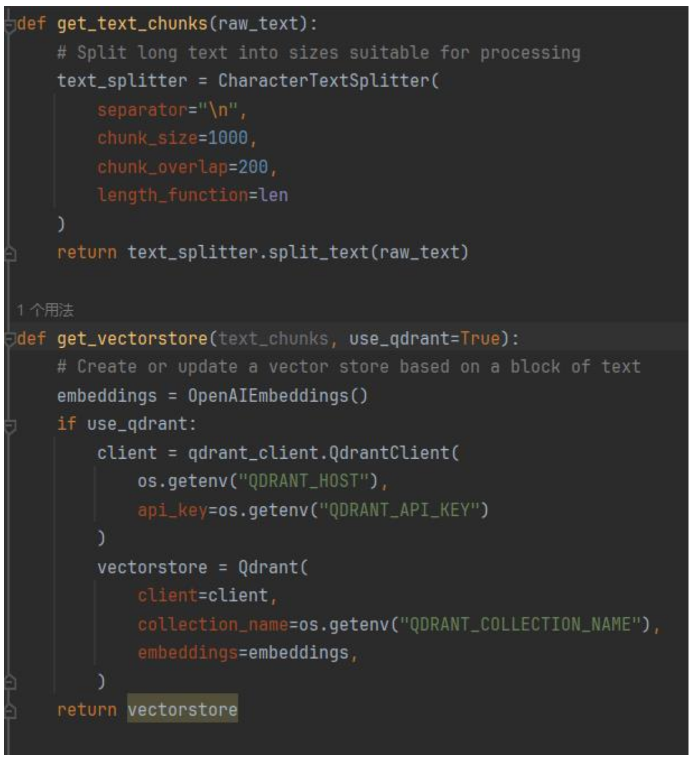
```
▾ [
  0 :
  "2024/2/28 12:13 HTML basics - Learn
  MDN
  https://developer.mozilla.org/en-
  US/docs/Learn/Getting_started_with_t
  1/11HTML basics
  HTML (HyperText Markup Language) is
  used to structure a web page and
  its content. For example, content co
  within a set of paragraphs, a list o
```

Chat with multiple PDFs 📖

Ask a question about your documents:

You can ask questions now.

Figure 5.2.5 Documents processing



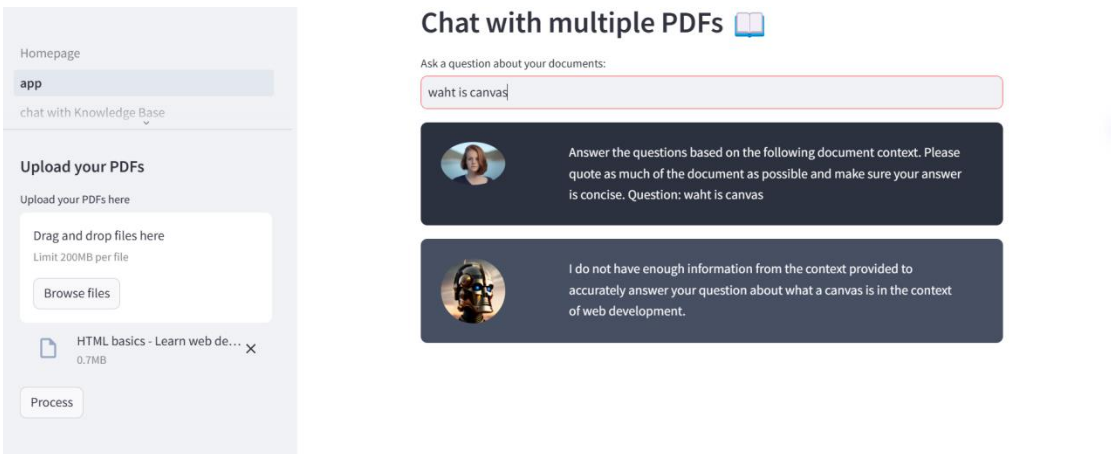
```python
def get_text_chunks(raw_text):
    # Split long text into sizes suitable for processing
    text_splitter = CharacterTextSplitter(
        separator="\n",
        chunk_size=1000,
        chunk_overlap=200,
        length_function=len
    )
    return text_splitter.split_text(raw_text)

1 个用法
def get_vectorstore(text_chunks, use_qdrant=True):
    # Create or update a vector store based on a block of text
    embeddings = OpenAIEmbeddings()
    if use_qdrant:
        client = qdrant_client.QdrantClient(
            os.getenv("QDRANT_HOST"),
            api_key=os.getenv("QDRANT_API_KEY")
        )
        vectorstore = Qdrant(
            client=client,
            collection_name=os.getenv("QDRANT_COLLECTION_NAME"),
            embeddings=embeddings,
        )
    return vectorstore
```

Figure 5.2.6 Key Code Snippets

Figure 5.2.7 shows how the application can answer questions based on the uploaded PDF. If there is no relevant answer in the PDF at the time of the question, the application will answer that he doesn't know, and he will only answer the question based on what is in the PDF. His code snippets are shown in Figure 5.2.8, Figure 5.2.9 and Figure 5.2.11.

In Figure 5.2.10, after receiving the question and PDF text content entered by the user, it is able to generate a complete answer and pass it to the session chain for processing. Once the processing is complete, the session history is displayed on the interface for the user to view. Its code snippet is in 5.2.9.

In Figure 5.2.12. this code snippet depicts the inclusion of a prompt template when the user interacts with the model in order to improve the accuracy of the model. By including such prompts at the time of interaction, the purpose of such prompts is to enhance the model's understanding of the user's intent and to improve the model's accuracy, resulting in more tailored responses.

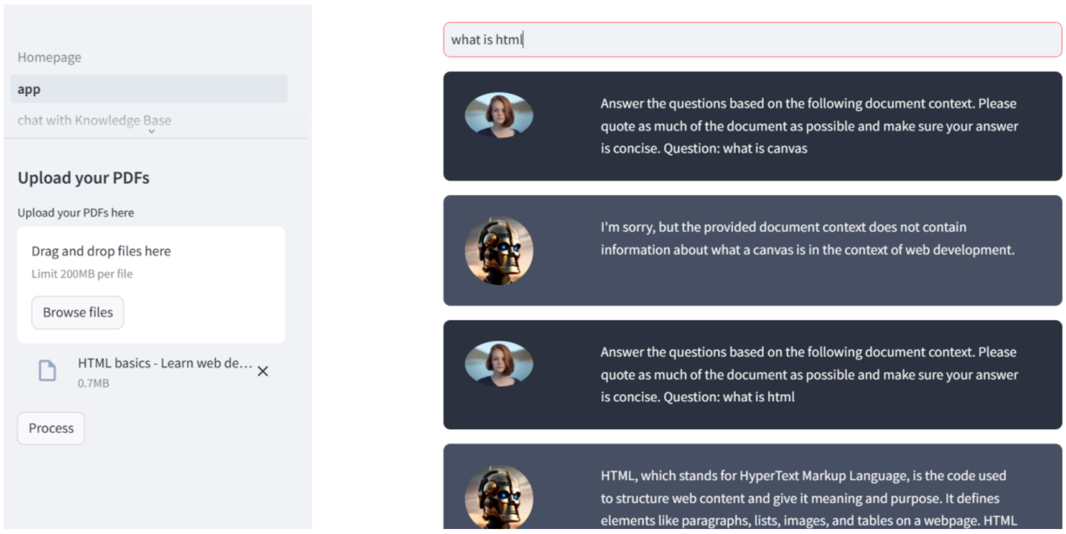


waht is canvas

Answer the questions based on the following document context. Please quote as much of the document as possible and make sure your answer is concise. Question: waht is canvas

I do not have enough information from the context provided to accurately answer your question about what a canvas is in the context of web development.

Figure 5.2.7：Answer the question



what is html

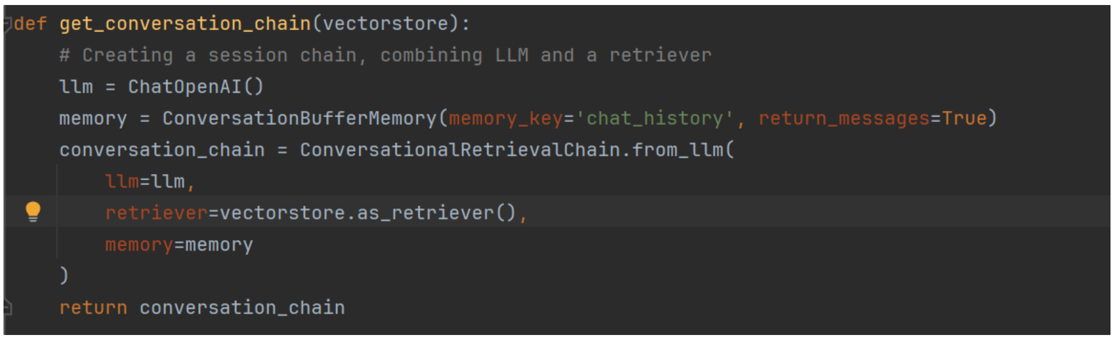
Answer the questions based on the following document context. Please quote as much of the document as possible and make sure your answer is concise. Question: what is canvas

I'm sorry, but the provided document context does not contain information about what a canvas is in the context of web development.

Answer the questions based on the following document context. Please quote as much of the document as possible and make sure your answer is concise. Question: what is html

HTML, which stands for HyperText Markup Language, is the code used to structure web content and give it meaning and purpose. It defines elements like paragraphs, lists, images, and tables on a webpage. HTML

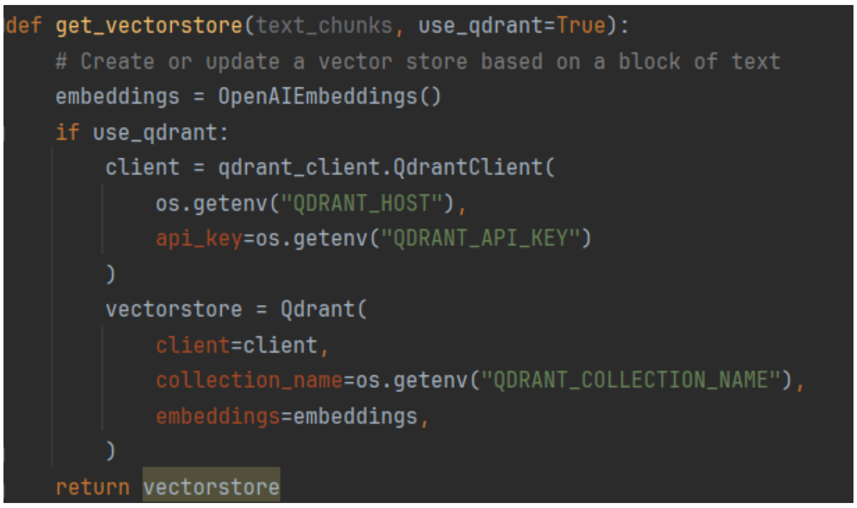
Figure 5.2.10 Historical content



```
def get_conversation_chain(vectorstore):
    # Creating a session chain, combining LLM and a retriever
    llm = ChatOpenAI()
    memory = ConversationBufferMemory(memory_key='chat_history', return_messages=True)
    conversation_chain = ConversationalRetrievalChain.from_llm(
        llm=llm,
        retriever=vectorstore.as_retriever(),
        memory=memory
    )
    return conversation_chain
```

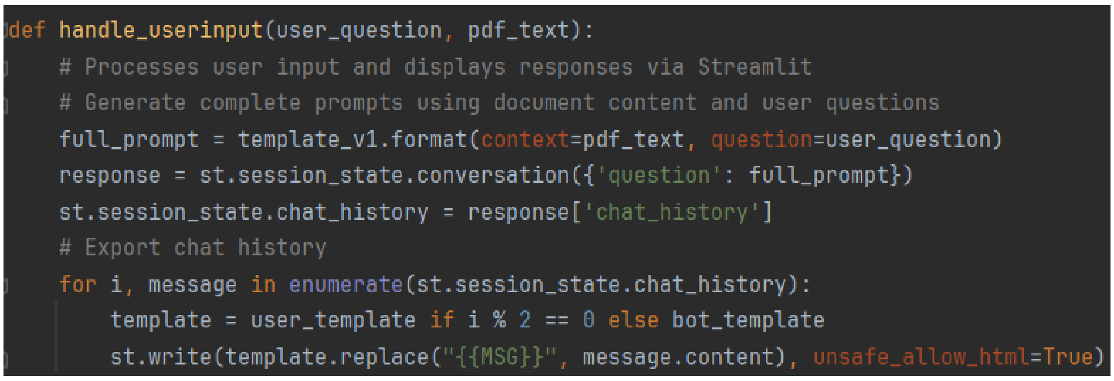
Figure 5.2.11

The key code snippet in Figure 5.2.7 is shown in Figure 5.2.8 Figure 5.2.9.



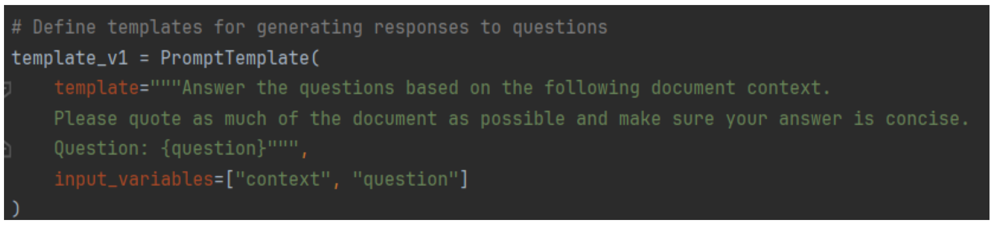
```python
def get_vectorstore(text_chunks, use_qdrant=True):
    # Create or update a vector store based on a block of text
    embeddings = OpenAIEmbeddings()
    if use_qdrant:
        client = qdrant_client.QdrantClient(
            os.getenv("QDRANT_HOST"),
            api_key=os.getenv("QDRANT_API_KEY")
        )
        vectorstore = Qdrant(
            client=client,
            collection_name=os.getenv("QDRANT_COLLECTION_NAME"),
            embeddings=embeddings,
        )
    return vectorstore
```

Figure 5.2.8



```python
def handle_userinput(user_question, pdf_text):
    # Processes user input and displays responses via Streamlit
    # Generate complete prompts using document content and user questions
    full_prompt = template_v1.format(context=pdf_text, question=user_question)
    response = st.session_state.conversation({'question': full_prompt})
    st.session_state.chat_history = response['chat_history']
    # Export chat history
    for i, message in enumerate(st.session_state.chat_history):
        template = user_template if i % 2 == 0 else bot_template
        st.write(template.replace("{{MSG}}", message.content), unsafe_allow_html=True)
```

Figure 5.2.9



```python
# Define templates for generating responses to questions
template_v1 = PromptTemplate(
    template="""Answer the questions based on the following document context.
    Please quote as much of the document as possible and make sure your answer is concise.
    Question: {question}""",
    input_variables=["context", "question"]
)
```

Figure 5.2.12

The dependent and outside libraries in Module 2 are the following:

Dotenv: This is a library for loading environment variables, implemented through the load_dotenv function. In this code, it is used to load environment variables such as Qdrant's host address and API key.

Streamlit: This is a library for building interactive web applications. In this code, Streamlit is used to create user interfaces, process user input, and display text and messages on web pages.

PyPDF2: This is a library for reading PDF files. In this code, PyPDF2 is used to extract text from uploaded PDF files.

Langchain: This is a library for natural language processing and may contain some text processing, embedding models and chat models. In this code, langchain is used to split long text, embed text and create dialogue models.

Qdrant_client: This is a library for interacting with the Qdrant vector retrieval service. In this code, qdrant_client is used to communicate with the Qdrant service, create vector stores and perform vector retrieval.

HtmlTemplates: This is an external library for defining HTML templates. In this code, htmlTemplates is used to define some web page styles or message templates.

The role of these dependencies and external libraries in the implementation is as follows:

Streamlining the development process: These libraries provide features and tools that can streamline the developer's workflow, such as reading PDF files, working with text data, building user interfaces, etc.

Provide core functionality: These libraries provide core functionality such as embedding models, dialogue models and vector retrieval services that are essential for implementing interactive chat applications.

Improve code reliability: Using these external libraries reduces the developer's effort in writing repetitive code and improves the reliability and maintainability of the code.

In summary, these dependencies and external libraries work together to build a fully functional interactive web application.

6. Testing

The focus of the testing phase was to evaluate the accuracy of the ChatGPT application's responses. The results showed that most of the responses generated by the application were accurate. However, since the application utilises the ChatGPT API, some inaccuracies and instances of "hallucinations" were observed, but these were not common. In cases where inaccuracies occurred, it was possible to ask further questions to clarify. If something was not included in the PDFs, the application informed that the information does not exist.

During the testing phase, we specifically tested whether the ChatGPT's responses were 100% accurate. The tests showed that the accuracy was not 100%. This system uses the GPT-3.5 model to generate responses. GPT-3.5 is a powerful language model, but it can sometimes make errors and is prone to hallucinations. Additionally, the system uses semantic search to find the most relevant text blocks, but it does not review the entire document. This means it might not find all relevant information or answer all questions, especially those that are summary-type or require extensive context from the documents. We also tested if files could be uploaded to the knowledge base and if they could be queried all at once, and the answer was affirmative.

In conclusion, for most use cases, the GPT-3.5 model is exactly accurate and can answer most questions effectively. It is essential to cross-check with the source materials to ensure the accuracy of the answers. The document upload feature was also performed very well.

7. Results and Evaluation

The application is currently deployed on the Streamlit public server. It can be accessed via the URL provided in my attached catalogue. However, as this is only a preliminary application and is not perfected, the application is currently in a non-public state. It will be made public in the future as the application is refined.

This project fulfils its initial goal of creating a ChatGPT application to teach students how to create a web (Html, CSS, JavaScript). During the testing phase, users reported that it would be better if we could develop a message function that would allow us to freely restore data from vector databases, and to freely modify the content, for which I will continue to focus on my knowledge of vector databases in the future. Also, if the answer is not 100 percent correct, in the future we will continue to pay attention to the latest and most accurate artificial intelligence technology, to achieve 100 percent accuracy.

In addition, compared to the traditional way of teaching by reading books, users reported that the use of AI and combined knowledge significantly improved efficiency.

8. Conclusions

This project successfully developed a ChatGPT-like application that leverages advanced AI technologies and a comprehensive PDF knowledge base to revolutionize the way students learn web development skills in HTML, CSS, and JavaScript. Through the implementation of LangChain and OpenAI's GPT models, the application provides real-time, interactive, and personalized learning experiences that are not just theoretical but relevant. The integration of AI in educational tools as demonstrated in this project shows significant promise in enhancing the effectiveness of learning and teaching, making education more accessible, engaging, and tailored to individual needs. The project has also highlighted several challenges and limitations, particularly related to the dependency on specific technologies and the scope of the AI's understanding based on the data provided in the PDF files. Nonetheless, the positive outcomes include improved student engagement and faster learning cycles, demonstrating the potential of AI to support and enhance traditional educational methodologies. Future work could focus on expanding the knowledge base, refining AI interactions, and exploring the application's adaptability to other programming languages and frameworks, thereby broadening its applicability and effectiveness in the educational sector. This conclusion reflects on the achievements and insights gained, as well as the prospective directions for further research and development in AI-driven educational technology.

9. Reference

1. Chen, L., Chen, P. and Lin, Z. (2020) Artificial Intelligence in Education: A Review, IEEE *Access*, 8, pp. 75264–75278. DOI:https://doi.org/10.1109/access.2020.2988510.

2. Zhang, Z. (2023) Underlying Logic and Possible Paths of ChatGPT/Generative Artificial Intelligence to Reshape Education, *Journal of East China Normal Universi*ty *(Education Science Edition)*, 41 (7), pp. 131. DOI:https://doi.org/10.16382/j.cnki.1000-5560.2023.07.012.

3. Tony (2023, May 19) Why We Need to Move Away from Traditional Learning Methods?, *Medium*. Medium. Available from: https://medium.com/@tonygreat1/why-we-need-to-move-away-from-traditional-learning-methods-1d1411dfe5c6 [Accessed 29 April 2024].

4. Jiao, J. (2023). *ChatGPT Boosting Digital Transformation of Education in Schooling. Chinese Journal of Distance Education,* (4). Available at: https://ms.jse.edu.cn/uploads/course/2023/06/20/16872585777521.pdf

5. Fang, X., Wang, X. and Ma, W., 2024. An Empirical Study on the Educational Application of ChatGPT. J. Electrical Systems, 20(2), pp.829-841.

6. Yuan, K., 2023. ChatGPT's Technology Application in the Higher Education Sector, Risk Analysis and Pathway Changes. Journal of Education and Educational Research, 3(3), pp.112-115.

7. Jafarzade, K., 2023. The Role of GPT Models in Education: Challenges and Solutions. 2023 5th International Conference on Problems of Cybernetics and Informatics (PCI), pp. 1-3. https://doi.org/10.1109/PCI60110.2023.10325940.

8. Nesbit, J., Adesope, O., Liu, Q., & Ma, W., 2014. How Effective are Intelligent Tutoring Systems in Computer Science Education? 2014 IEEE 14th International Conference on Advanced Learning Technologies, pp. 99-103. https://doi.org/10.1109/ICALT.2014.38.

9. VanLehn, K., 2011. The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems. Educational Psychologist, 46, pp. 197 - 221. https://doi.org/10.1080/00461520.2011.611369.

10. Steenbergen-Hu, S., & Cooper, H., 2014. A meta-analysis of the effectiveness of intelligent tutoring systems on college students' academic learning. Journal of Educational

Psychology, 106, pp. 331-347. https://doi.org/10.1037/A0034752.

11. Zhang, D., 2005. Interactive Multimedia-Based E-Learning: A Study of Effectiveness. American Journal of Distance Education, 19, pp. 149 - 162. https://doi.org/10.1207/s15389286ajde1903_3.

12. Shen, R., Wohn, D., & Lee, M., 2020. Programming Learners' Perceptions of Interactive Computer Tutors and Human Teachers. Int. J. Emerg. Technol. Learn., 15, pp. 123-142. https://doi.org/10.3991/ijet.v15i09.12445.

13. Khan, M.J., Omar, M. & Jian, J., 2023. Personalized learning through AI. Journal of AI Research, 5(1), pp. 1-19. Available at: https://doi.org/10.54254/2977-3903/5/2023039

14. Murtaza, M., Ahmed, Y., Shamsi, J., Sherwani, F., & Usman, M., 2022. AI-Based Personalized E-Learning Systems: Issues, Challenges, and Solutions. IEEE Access, 10, pp. 81323-81342. https://doi.org/10.1109/access.2022.3193938.

15. Z.Zhu,(2021). Composition of Online Teaching and Academic Ability under the Background of Artificial Intelligence and HTML. In 5th International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 2021, pp. 1467-1470, doi: 10.1109/ICCMC51019.2021.9418250.

16. Jumankuziev Uktamjon (2023). THE ROLE OF TEACHERS IN TEACHING PROGRAMMING LANGUAGES IN HIGHER EDUCATIONAL INSTITUTIONS OF PEDAGOGY. Gospodarka i Innowacje., [online] 41, pp.360–362. Available at: https://www.gospodarkainnowacje.pl/index.php/issue_view_32/article/view/1948 [Accessed 29 Mar. 2024].

17. Pan, J.J., Wang, J. and Li, G. (2023). Survey of Vector Database Management Systems. [online] arXiv.org. Available at: https://arxiv.org/abs/2310.14021 [Accessed 23 Mar. 2024].

18. Alejandro AO (2023). Chat with Multiple PDFs | LangChain App Tutorial in Python (Free LLMs and Embeddings [Online video]. Available at: https://www.youtube.com/watch?v=dXxQ0LR-3Hg (Accessed: 15 January 2024).

19. Alejandro AO. (2023). Langchain + Qdrant Cloud | Pinecone FREE Alternative (20GB) | Tutorial. YouTube. Available at: https://www.youtube.com/watch?v=VL6MAAgwSDM [Accessed 7 Mar. 2024].

10. Appendices

Application deployment URL

https://chat-with-pdf12.streamlit.app/

Link to upload files

https://colab.research.google.com/drive/1z1uCOtmjRxfXJdNEIMpH5beFpvF5f75_?usp
=sharing