WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Quillchat

Simran Galwani[1], Atharva Kute[2], Piyush Pradhan[3], Pragati Pandhare[4], Mr. Tejas Shah[5]

Department of Computer Engineering, Vivekanand Education Society's Polytechnic, Chembur, Mumbai-71

ABSTRACT :

In today's digital age, accessibility to information is most important. However, many documents, especially those in Portable Document Format (PDF), pose challenges for users with varying abilities to access, comprehend, and interact with the content. To address this issue, we propose the development of a Quillchat, an innovative solution that using natural language processing (NLP) and machine learning techniques to facilitate seamless interaction with PDF documents. Our project aims to design and implement a quillchat capable of parsing and extracting relevant information from PDF documents upon user request. Through the integration of advanced NLP algorithms, the quillchat will understand user queries, extract relevant data from the PDF, and provide concise and accurate responses in real-time. Additionally, the quillchat will offer functionalities such as summarization, keyword extraction, and navigation assistance to enhance the user experience further.

Keywords: Adaptability, robustness, scalability.

INTRODUCTION :

Quillchat is a type of online platform that helps users to interact with multiple pdf's at a time and extract data more quickly and effortlessly.

In an increasingly digital world, the prevalence of PDF documents for sharing information is undeniable, yet managing and extracting relevant data from multiple PDFs remains a challenging task. Our project aims to address this challenge by developing a quillchat, an AI-driven conversational interface designed to assist users in efficiently handling and extracting information from numerous PDF documents. PDF files are commonly used for document sharing and reading, but they may often be difficult to work with. Fortunately, there are many AI-powered tools that can help you interact with PDF files in a more convenient and intuitive way. UPDF, LightPDF, and PDFGPT are some examples of online tools that use AI to provide various PDF functionalities, such as summarizing, translating, editing, converting, compressing, signing, and more. These tools can save you time and effort while working with PDFs. Key objectives include designing a robust quillchat system capable of simultaneously managing multiple PDF documents, integrating advanced natural language processing (NLP) algorithms for seamless user interaction, and implementing features such as document clustering and summarization to enhance efficiency. The quillchat will provide users with the ability to query, navigate, and extract insights from multiple PDFs through a user-friendly interface. Our approach involves data preprocessing to extract text and metadata from PDFs, NLP integration for understanding user queries and document contents, and the development of a scalable chatbot architecture. Additionally, we prioritize accessibility features to ensure inclusivity for users of all abilities. Expected outcomes of the project include the development of a functional Quillchat prototype, improved efficiency in managing large volumes of PDF documents, and a demonstration of the effectiveness of AI-driven chatbot technology in document management tasks. Ultimately, the project aims to contribute to advancements in document processing, AI-powered chatbots, and accessibility in information access.[1]

ADVANTAGES OF QUILLCHAT :

1. **Simple and interactive user interface:** Users can effortlessly interact with the pdfs which are uploaded by the users itself.
2. **Document parsing and analysis:** Developed a quillchat leveraging PDF parsing libraries to extract text from documents, and applied NLP techniques for analysis such as summarization, and sentiment analysis, enabling users to interact and query information within PDF documents seamlessly.
3. **Document summarization:** Integrate PDF parsing capabilities with summarization algorithms to extract key information from documents, enabling the quillchat to provide concise summaries tailored to user queries, enhancing accessibility and efficiency in information retrieval.
4. **Query based document retrieval:** Leverage PDF parsing to extract relevant information based on user queries, utilizing natural language processing techniques to enable the quillchat to retrieve specific sections or pages from documents, streamlining access to desired content within PDF files.
5. **Data extraction:** Utilize PDF parsing techniques to extract structured data such as text, tables, and metadata from PDF documents, enabling

the quillchat to efficiently retrieve and present relevant information to users, facilitating seamless data access and analysis.[2]

LITERATURE SURVEY :

A literature survey for Quillchat would entail exploring three key domains: natural language processing (NLP), question answering systems, and document processing. In NLP, recent advancements in deep learning and transformer-based models like BERT and GPT have revolutionized the field by enabling machines to understand and generate human-like text. Research papers and articles in this area can provide insights into leveraging these techniques for tasks such as text summarization, sentiment analysis, and named entity recognition, all of which are relevant to QuillChat's functionality.
Question answering systems form another critical area of investigation. Studies on state-of-the-art QA models, including those trained on large-scale datasets like squad, can inform the development of QuillChat's ability to extract relevant information from documents and provide accurate responses to user queries. Additionally, understanding approaches for document retrieval, passage ranking, and summarization is essential for enhancing the system's performance in handling unstructured text data. Finally, research in document processing is necessary for QuillChat's ability to extract data from PDF files and other document formats. This involves exploring OCR techniques, document layout analysis, and tools/libraries for efficient text extraction. By reviewing literature in these domains and incorporating data from existing research, developers can ensure that Quillchat leverages state-of-the-art techniques and best practices to deliver a robust and effective document-oriented chatbot experience for users.
Technological Foundations

- **PDF Text Extraction:**
  The code can read and extract text from PDF documents that users upload. It ensures that the content of the PDFs becomes accessible for further interaction.
- **Text Chunking:**
  To manage and process large amounts of text more efficiently, the extracted content is divided into smaller, manageable chunks. This helps in handling information more effectively during conversations.
- **Google Generative AI Embeddings:**
  The application utilizes Google Generative AI embeddings to convert the chunks of text into numerical representations. These embeddings capture the meaning of the text, enabling the quillchat to understand and respond contextually.
- **FAISS Vector Store:**
  The code employs FAISS to create a vector store from the embeddings. This vector store facilitates rapid and efficient similarity searches, ensuring the chatbot can quickly find relevant information.
- **Conversational Chain Model:**
  The code integrates a conversational chain model, powered by ChatGoogleGenerativeAI, allowing the quillchat to engage in more natural and context-aware conversations. It considers both the provided context and the user's question in generating responses.
- **Chat Interface with Streamlit:**
  The user interacts with the quillchat through a user-friendly interface created using Streamlit. Users can upload PDFs, ask questions in a chat-like format, and receive responses in real-time.
- **Chat History Management:**
  The application keeps track of the conversation, allowing users to review past interactions. The "Clear Chat History" button provides a convenient way to reset the chat and start fresh.
- **Word Document Saving:**
  Users have the option to save the entire chat history, including questions and answers, to a Word document. This feature enhances user convenience by providing a record of the conversation for future reference.

3. SYSTEM IMPLEMENTATION :

A. EXPERIMENTAL SETUP

Visual Studio code
Visual Studio Code (VS Code) is a popular, free, open-source code editor developed by Microsoft. It's available for Windows, macOS, and Linux. VS Code is widely used by developers for a variety of programming languages, including but not limited to JavaScript, TypeScript, Python, PHP, C++, and C#. It's known for its performance, versatility, and the extensive ecosystem of extensions that enhance its functionality.
Key features of Visual Studio Code include:
IntelliSense: Provides smart completions based on variable types, function definitions, and imported modules.
Debugging: Built-in debugging support that can launch or attach to your running apps and debug with breakpoints, call stacks, and an interactive console.
Extensions: A vast marketplace of extensions to add languages, debuggers, and tools to your installation to support your development workflow.
Git Integration: Built-in Git support to review diffs, stage files, and make commits right from the editor. You can also push and pull from any hosted Git service.
Customization: Highly customizable, allowing users to change themes, keyboard shortcuts, preferences, and install extensions to add additional functionality.

Portable Mode: Ability to run it from a USB stick or other portable storage device, making it easy to carry your settings and extensions with you.
VS Code also supports tasks and snippets, includes a built-in terminal, offers syntax highlighting for a wide range of programming languages, and provides a rich API for developers to build their own extensions. Its lightweight nature, combined with powerful features, makes it a preferred choice for coding, app development, and web development tasks.[3]

LangChain :

LangChain is a framework for developing applications powered by language models. It enables applications that:
Are context-aware: connect a language model to sources of context (prompt instructions, few shot examples, content to ground its response in, etc.)
Reason: rely on a language model to reason (about how to answer based on provided context, what actions to take, etc.)
This framework consists of several parts.

Components: Langchain refers to a hypothetical system integrating blockchain technology with language processing components for decentralized language learning, translation, or communication.
Purpose: To facilitate decentralized language learning, translation, and communication through the integration of blockchain technology and language processing components.
Ease of Use: Intuitive interface and seamless integration of blockchain technology, simplifying decentralized language learning, translation, and communication.
Scalability: Its robust architecture, allowing for seamless expansion to accommodate growing user demands and linguistic interactions on a decentralized network.
Cross-Platform: Enables seamless access and interaction across various devices and operating systems, facilitating widespread adoption and user engagement.[4]

Python
Python is a general-purpose programming language that is used for web development, software development, machine learning, and data science. It is an interpreted language, which means that it does not need to be compiled before it can run. Python is also a dynamically typed language, which means that the type of a variable is not declared until it is assigned a value.

Components: Its core language, standard library, and various third-party modules for diverse functionalities, all contributing to its versatility and widespread usage.
Purpose: To provide a versatile, readable, and efficient programming language suitable for various applications, ranging from web development and data analysis to artificial intelligence
Ease of Use: Its clear and concise syntax, extensive documentation, and large supportive community, making it accessible for both beginners and experienced developers alike.
Scalability: Handle projects of varying sizes, from small scripts to large-scale applications, owing to its flexibility, extensive ecosystem, and support for modular design.
Cross-Platform: Python's cross-platform nature enables seamless execution on diverse operating systems such as Windows, macOS, and Linux, ensuring consistent behavior and development experience across different environments.[5]

B. PROJECT PROCEDURE AND FLOW :

- To develop this project efficiently, communicate with the project guide and also a couple of corporate developers.
- First, finalize the features and specifications which shall be implemented in our project.
- After having a clear vision of features design the web application with an intention to have a decent UI UX.
- This includes thinking of where the button shall be placed, a click on the button should display which page, to summarize in short, it aims to make sure the user has a convenient and efficient user experience
- Once the design is ready and approved, start with the development of the actual project.
- Once the web application is developed and ready, proceed with one of the important things of software development life cycle-testing
- If any bugs or defects are found, they would be resolved and retested to gain confidence in the project.

C. MODELLING AND ANALYSIS

- Data flow diagram (DFD)

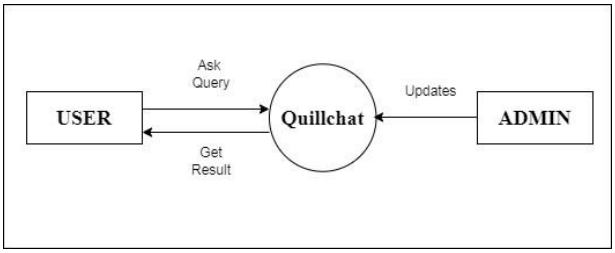

Figure1: DFD Level 0

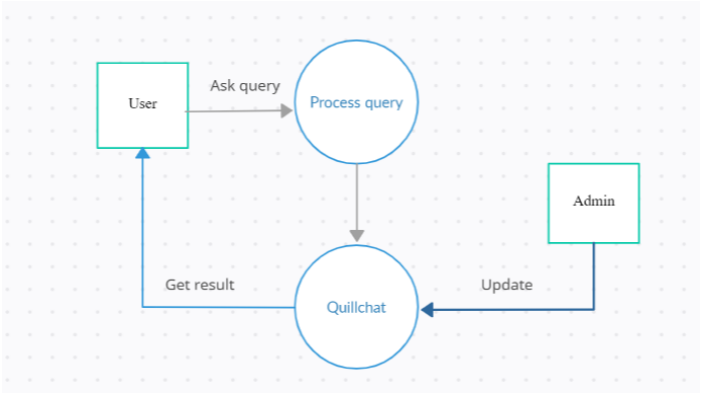

Figure2: DFD Level 1

- **Use Case Diagram**

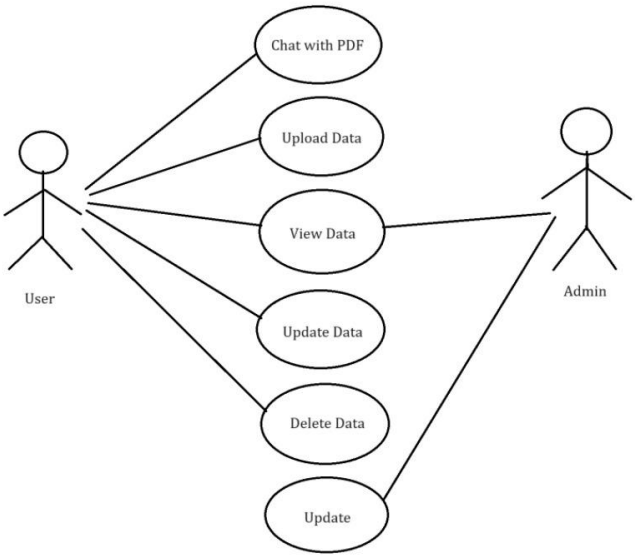

Figure3: Use case Diagram

D. PROJECT ARCHITECTURE FLOW

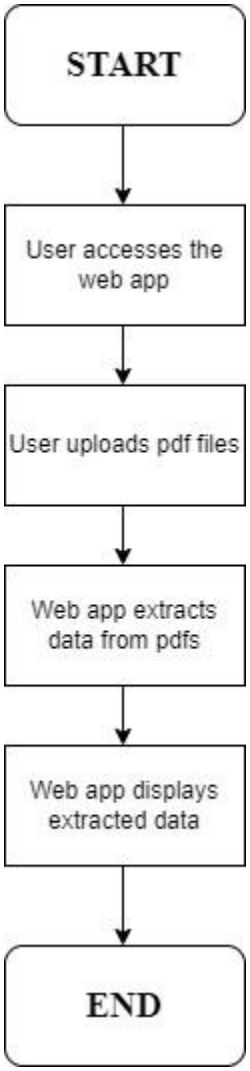

Figure1: Project Architecture Flow

- **User Interface**

4. FUTURE SCOPE :

The future scope of quillchat is bright and promising, as we have planned to add some more amazing and useful features to our applications.

i) Copy Button: This feature enhances the user's ability to utilize the extracted information. After receiving a response to a query, the application will display a copy button. Upon clicking, the entire interaction, including the user's question and the chatbot's answer, will be copied. Users can then paste this content directly into a word document within the application, streamlining the process of compiling information.

ii) PDF Preview section: The PDF preview section acts as a visual aid to the user's interactions. When a question is posed, the application not only provides a textual response but also highlights the corresponding section in the original PDF from which the information was extracted. This visual cue helps users quickly locate and verify the context of the information within the source document.

iii) Keyword highlighter: This feature enhances the user's ability to identify key terms within the document. When a question is asked, the application will not only provide a response but will also dynamically highlight the exact keyword or keywords in the PDF that were instrumental in generating the answer. This assists users in understanding the context and relevance of the highlighted terms within the source material.

iv) OCR: Optical Character Recognition(OCR) technology expands the application's capabilities beyond traditional PDFs. In addition to uploading PDF documents, users can now leverage OCR to extract information from images. The application employs OCR to convert text within images into machine-readable text, enabling users to pose questions based on image content. This functionality significantly broadens the types of documents that users can interact with, making the application more versatile and inclusive.

5. Conclusion:

Quillchat is useful tool for managing and finding information in PDF documents. We have used smart technology to understand what you're looking for and help you quickly locate the right information without wasting time scrolling through pages. By making it easy to access important details, quillchat make work more efficient and help you make better decisions.

Looking forward, Quillchat will keep getting better, becoming even more helpful as they learn from user interactions and adapt to individual needs. As technology improves, quillchat will become essential for anyone dealing with lots of documents, making it easier than ever to find what you need when you need it.

REFERENCESS :

[1] https://www.geeksforgeeks.org/
[2] https://www.geeksforgeeks.org/
[3] https://code.visualstudio.com/
[4] https://www.langchain.com/
[5] https://www.python.org/