**SmartPDF-Chat Project Report**

**1. Introduction**

**1.1 Overview**

SmartPDF-Chat is a web application designed to function as a conversational interface for working with PDFs. Users can upload PDFs, specify relevant page ranges, and then interact with the document content through a chat-like interface. Unlike traditional PDF search methods, SmartPDF-Chat focuses on providing highly relevant answers to user questions based on the context of the uploaded documents.

**1.2 Motivation and Objectives**

The primary motivation behind SmartPDF-Chat is to enhance the user experience when working with PDFs. Traditional methods of searching and extracting information from PDFs can be time-consuming and cumbersome. SmartPDF-Chat aims to simplify this process by offering a conversational approach, allowing users to ask questions and receive answers directly related to the uploaded documents.

**2. Problem Statement**

SmartPDF-Chat addresses the specific challenge of context-aware information retrieval in PDFs. Here's a breakdown of the problems it solves:

* Traditional PDF search functionalities often lack context, leading to irrelevant results that don't answer the user's specific questions.
* Manually skimming through multiple PDFs to find specific details is time-consuming and error-prone. This can be frustrating and inefficient.

**3. Objectives**

SmartPDF-Chat addresses these issues by providing the following functionalities:

* Enables users to ask natural language questions about the content within the uploaded PDFs.
* Employs Natural Language Processing (NLP) techniques to understand the user's intent and the context of their questions.
* Searches across multiple PDFs and pinpoints relevant passages that directly address the user's query.
* Utilizes a large language model (LLM) to process the retrieved information and generate focused answers based on the overall context of the uploaded PDFs.

**4. Scope**

SmartPDF-Chat focuses on the following functionalities:

* Uploading and processing text-based PDFs. It currently cannot handle non-textual content like images, tables, or encrypted PDFs protected by passwords.
* Understanding user queries phrased in natural language.
* Retrieving and summarizing information from the specified page ranges within uploaded PDFs.

**5. Methodology**

SmartPDF-Chat utilizes the following core functionalities to achieve its objectives:

* **PDF Processing:** Parses text content from uploaded PDFs using libraries like PyPDF2.
* **Natural Language Processing (NLP):**
  + **Text Segmentation:** Splits the extracted text into manageable chunks for further processing.
  + **Text Embeddings:** Creates numerical representations of text chunks using pre-trained models like Sentence Transformers. This helps the system understand the relationships between words and concepts.
  + **Information Retrieval:** Identifies relevant passages within the PDFs based on user queries using techniques like vector similarity search. This involves comparing the user's query to the numerical representations of the text chunks to find the most relevant passages.
  + **Question Answering:** Employs a Retrieval-QA approach, combining information retrieval with the LLM from the to answer user questions in context.
* **Large Language Model (LLM):** Utilizes the **togethercomputer/llama-2-70b-chat** model specifically to process the retrieved information, generate summaries of the relevant passages, and formulate responses to user queries in a comprehensive way. This LLM is chosen for its focus on conversation and its ability to understand and respond to natural language questions.

**6. Expected Outcomes**

The expected outcomes of SmartPDF-Chat are enhanced to reflect the focus on context-aware Q&A:

* Users can efficiently find answers to their specific questions within the context of multiple PDFs. SmartPDF-Chat will significantly reduce the time and effort required to find the information they need.
* SmartPDF-Chat reduces the time and effort required to extract information from complex documents. Users can ask questions directly instead of manually searching through pages of text.
* Answers are tailored to the user's query and the broader context of the uploaded PDFs. The system will not just provide isolated snippets of text but will consider the surrounding information to provide a comprehensive and informative answer.

**7. Timeline**

The development of SmartPDF-Chat can be divided into the following phases:

* Phase 1 (1 week): Design and development of the core functionalities – PDF processing, text segmentation, and information retrieval.
* Phase 2 (1 week): Integration of the Together LLM platform and trying out various LLM that it offers and choosing the one that satisfy our needs.
* Phase 3 (1 week): User interface development and testing. This phase involves creating an interface where users can upload PDFs, ask questions, and receive answers. Streamlit is a suitable web framework for this purpose.
* Phase 4 (1 week): Documentation and deployment. This phase involves creating documentation for the project, including instructions on how to use the application and how it works. Additionally, the application will be deployed to a web server so that users can access it remotely.

**8. Resources Required**

* **Hardware:** Computer with at least 16 GB RAM and a 4GB GPU. The GPU will accelerate the NLP tasks, especially when dealing with large or complex PDFs.
* **Software:**
  + Python programming language: Python is a versatile language commonly used for data science and machine learning applications.
  + Streamlit web framework for developing the user interface: Streamlit allows for the creation of web apps in Python with minimal coding.
  + PyPDF2 library for PDF processing: PyPDF2 is a Python library specifically designed for working with PDF files.
  + Langchain library for NLP functionalities: Langchain provides a suite of tools for NLP tasks like text processing and information retrieval.
  + Together LLM platform: Provides access to the **togethercomputer/llama-2-70b-chat** large language model.
* **Datasets:** Pre-trained NLP models for text embeddings. These models are essential for creating numerical representations of text, which is a crucial step in information retrieval.

**9. References**

* Streamlit Documentation: <https://docs.streamlit.io/>
* PyPDF2 Documentation: <https://pypdf2.readthedocs.io/>
* Langchain: <https://www.langchain.com/>
* Together LLM platform: <https://docs.together.ai/docs/quickstart> (Focuses on using the platform to access the LLM)
* Sentence Transformers: <https://huggingface.co/sentence-transformers>
* Hugging Face: <https://huggingface.co/>