Project Report: PDF Upload and Question-Answering Web Application

ANUJ CHAUDHARY 21112028

Introduction

This project involves the creation of a web application that allows users to upload PDF documents and ask questions based on the content of these documents. The application extracts text from the uploaded PDF, processes it, and uses a pre-trained question-answering model to provide answers to user queries. The project leverages Flask for the web framework, PyMuPDF for PDF text extraction, and the Transformers library from Hugging Face for the question-answering model.

Objective

The primary objective of this project is to provide a seamless interface for users to upload PDF documents and retrieve information from them through natural language questions. This functionality is useful in various contexts such as academic research, business document analysis, and legal document review.

Technologies Used

- **Flask**: A lightweight WSGI web application framework in Python.
- **PyMuPDF (fitz)**: A Python binding for MuPDF to extract text from PDF files.
- **Transformers (Hugging Face)**: A library that provides general-purpose architectures for NLP including the question-answering model.
- **HTML/CSS**: For creating a user-friendly frontend interface.

Implementation Details

Directory Structure

The project directory contains the following key files:

- `app.py`: The main Flask application script.
- `templates/index.html`: The HTML template for the web application.
- `static/styles.css`: The CSS file for styling the web application.

- `docs/`: Directory to store uploaded PDF files.

Flask Application (app.py)

The Flask application handles the backend logic, including routes for rendering the homepage, uploading PDF files, and processing user queries.

HTML Template (index.html)

The HTML file provides the structure and layout of the web interface. It includes a form for uploading PDFs and a text input for asking questions.

Workflow

1. **Uploading PDF**: Users can upload a PDF file using the provided form.
2. **Extracting Text**: The application reads the text content from the uploaded PDF using PyMuPDF.
3. **Asking Questions**: Users can input questions related to the content of the PDF.
4. **Generating Answers**: The question-answering model processes the question and context to generate an answer.
5. **Displaying Results**: The answer is displayed on the web page.

Conclusion

This project successfully demonstrates the integration of web technologies and machine learning models to create an interactive question-answering application based on PDF content. The use of Flask, PyMuPDF, and Transformers ensures a seamless experience for users to upload documents and retrieve specific information through natural language queries.

Future improvements could include enhanced error handling, support for more file formats, and improvements in the UI/UX to make the application more user-friendly and robust.