AI Agent for Multiple-Choice Question Generation from PDF

1. Introduction

This document outlines the implementation of an AI-powered system that generates multiple-choice questions from a book PDF and saves them to a database. The system includes modules for PDF text extraction, question generation using NLP models, and storage in a structured database.

2. Tech Stack & Tools Required

- Programming Language: Python (for AI and backend processing)
- PDF Extraction: PyPDF2 or pdfminer.six
- NLP & Question Generation: OpenAI's GPT model or transformers from Hugging Face
- Database: MySQL or MongoDB
- Optional Frontend: React.js for a user interface

3. Architecture Overview

1. PDF Upload & Extraction Module: Extracts and preprocesses text from the uploaded PDF.
2. Question Generation Module: Uses NLP models to generate questions and distractors.
3. Database Integration Module: Stores questions, options, and metadata in a database.
4. Optional User Interface (UI): Allows users to upload PDFs, review questions, and manage the question bank.

Step 1: PDF Extraction

Use pdfminer.six or PyPDF2 to extract text from the PDF. Clean and preprocess the text by removing special characters, extra spaces, and page numbers.

Example Code:

```
from pdfminer.high_level import extract_text

def extract_text_from_pdf(file_path):

    text = extract_text(file_path)
    cleaned_text = text.replace('\n', ' ').replace('\t', ' ')

    return cleaned_text

pdf_text = extract_text_from_pdf('path/to/book.pdf')
```

Step 2: Question Generation

Use OpenAI's GPT model or Hugging Face's transformers to generate questions. Split the text into smaller chunks to provide context for each question. Generate a question and multiple-choice options for each chunk.

Step 3: Store Questions in Database

Choose between MySQL or MongoDB for structured or flexible storage. Example schema and code for MySQL using SQLAlchemy are provided.

Step 4: Optional Frontend (React.js)

- Upload PDF: UI for uploading PDF files.

- Question Review: Display generated questions for review and editing.

- Question Management: Edit, delete, or categorize questions.

Step 5: Deployment

- Backend: Deploy using Docker or cloud services like AWS, Azure, or Google Cloud.

- Frontend: Deploy using Netlify, Vercel, or similar platforms.

- Database: Use cloud-hosted solutions like AWS RDS (MySQL) or MongoDB Atlas.

6. Further Enhancements

- Advanced NLP for better question quality.

- Auto-categorization and difficulty classification.

- Admin Dashboard for managing the question bank.

- Reporting and analytics for question performance.