Product Requirements and Specification Document

Project Name

PDFQuery - PDF-based Question Answering System

Description

PDFQuery is an open-source tool for the education sector that enables users to upload PDF documents and ask questions about their content. Answers are generated using Retrieval-Augmented Generation (RAG) techniques via LangChain and OpenAI APIs. The system is built with Python and Flask.

---

1. Goals & Objectives

Goal	Description
Enable PDF-based Q&A	Users can upload PDFs and ask questions about their content
Leverage RAG with LangChain & OpenAI	Use state-of-the-art generative AI for accurate answers
Simple, intuitive UI	Minimal learning curve for educators and students
Open-source & extensible	Codebase is public and easy to extend

---

2. Target Users

User Type	Needs
Educators	Quickly extract and explain PDF content
Students	Ask questions and clarify PDF material
Developers	Extend and integrate the tool

---

3. Core Features

Feature	Description
PDF Upload	Users upload one or more PDF files
Question Input	Users enter natural language questions
RAG-based Answering	System retrieves relevant PDF content and generates answers via OpenAI
Answer Display	Answers are shown with source context snippets
History	Session-based Q&A history for user reference
Open-source Repository	Public codebase with documentation

4. Functional Requirements

ID	Requirement
FR1	Users can upload PDF files (max 20MB per file, up to 3 files per session)
FR2	Users can input questions in natural language
FR3	System extracts and indexes PDF text upon upload
FR4	System retrieves relevant text chunks using LangChain
FR5	System generates answers using OpenAI's API (e.g., GPT-3.5/4)
FR6	Answers are displayed with referenced PDF context
FR7	Session Q&A history is maintained until browser refresh
FR8	Basic error handling for file type, size, and API failures

5. Non-Functional Requirements

ID	Requirement
NFR1	Response time < 10 seconds per question
NFR2	Secure file handling and data privacy
NFR3	Modular, well-documented codebase
NFR4	Compatible with modern browsers
NFR5	Easy local deployment (Docker support)

6. Technical Specifications

Component	Technology / Approach
Backend	Python 3.x, Flask
PDF Processing	PyPDF2 or pdfplumber
RAG Pipeline	LangChain (retrieval, chunking, embedding)
LLM Integration	OpenAI API (configurable model)
Frontend	Flask templates (Jinja2), Bootstrap for UI
Storage	In-memory (session-based), no persistent storage
Deployment	Dockerfile, requirements.txt
Open Source	MIT License, GitHub repository

7. User Flow

```
flowchart TD
    A[Upload PDF(s)] --> B[PDF Text Extraction & Indexing]
    B --> C[User Inputs Question]
    C --> D[Retrieve Relevant Chunks (LangChain)]
    D --> E[Generate Answer (OpenAI)]
    E --> F[Display Answer & Context]
    F --> G[Show Q&A History]
```

---

8. API & Integration

Endpoint	Method	Description
/upload	POST	Upload PDF(s)
/ask	POST	Submit question, receive answer
/history	GET	Retrieve session Q&A history

---

9. Security & Privacy

- Uploaded PDFs processed in-memory; not stored after session ends
- API keys managed via environment variables
- Input validation and file type checks

---

10. Open Issues & Future Enhancements

Area	Notes / Future Work
User Authentication	Optional for future versions
Persistent Storage	Optional for Q&A history retention
Multi-language	Support for non-English PDFs/questions
Advanced UI	Richer interface, highlighting, PDF preview

---

11. Acceptance Criteria

- Users can upload PDFs and ask questions; answers reference PDF content
- System responds within 10 seconds per query
- No user data or files persist after session ends
- Codebase is open-source, documented, and Dockerized

---

12. Out of Scope

- OCR for scanned PDFs

- Real-time collaboration
- Mobile app

---

13. Milestones

Milestone	Target Date
MVP Feature Complete	4 weeks
Open Source Release	5 weeks
Documentation Complete	5 weeks

End of Document