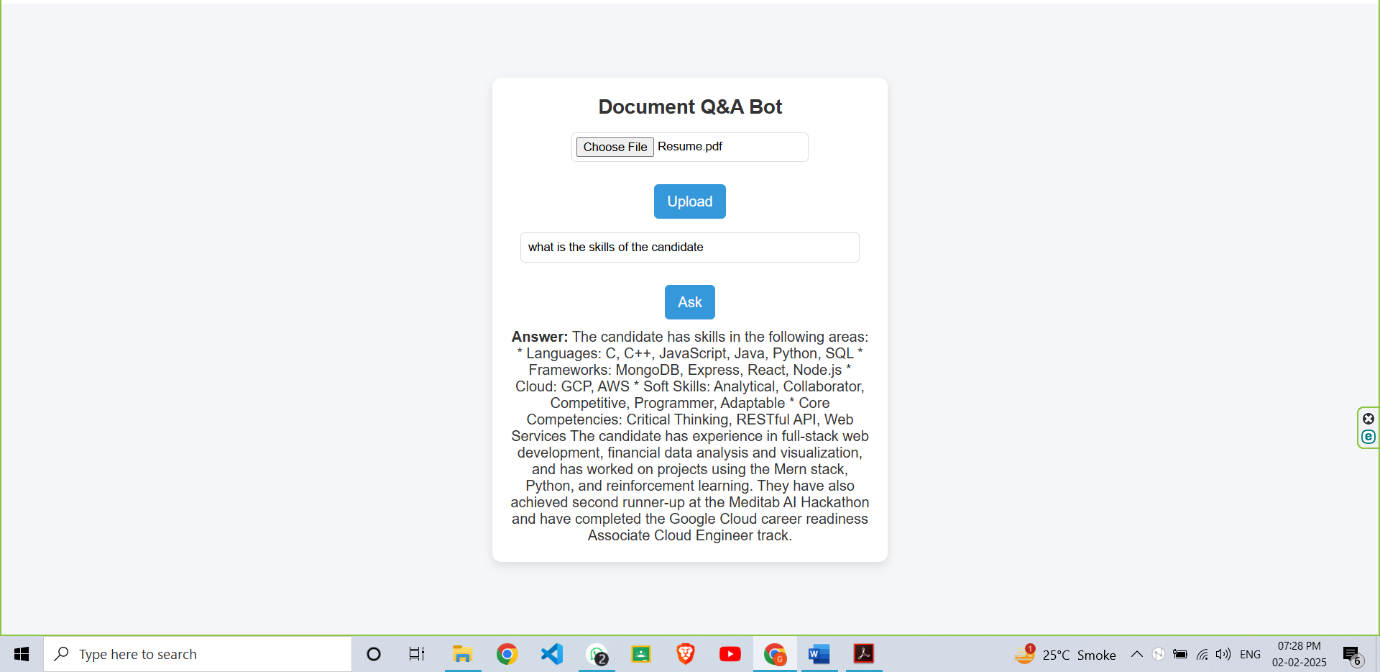
# Document Q&A Bot Documentation

## Project Overview

The Document Q&A Bot is a full-stack web application that allows users to upload documents (PDFs and text files) and ask questions based on the content of those documents. The application leverages an AI model to generate answers to the user's questions. This project is built using the MERN stack (MongoDB, Express, React, Node.js) with pdf-parse for handling PDFs.



## Directory Structure

Copy

priyanshu6091-document\_qna/

├── README.md

├── backend/

│ ├── index.js

│ ├── package.json

│ └── ...

└── frontend/

├── eslint.config.js

├── index.html

├── package.json

├── vite.config.js

├── public/

└── src/

├── App.css

├── App.jsx

├── index.css

├── main.jsx

└── ...

## Features

* **Upload PDF and text files**
* **Ask questions based on the uploaded document's content**
* **Get AI-generated answers to the questions**
* **Real-time error handling and validation**
* **User-friendly, modern UI with Tailwind CSS**
* **Responsive Design for all devices**

## Tech Stack

| **Technology** | **Usage** |
| --- | --- |
| **Frontend** | React, Tailwind CSS, Vite, Axios |
| **Backend** | Node.js, Express.js, Multer |
| **PDF Handling** | pdf-parse |
| **Storage** | MongoDB (if needed for user history) |
| **Deployment** | Vercel (Frontend), Render (Backend) |

## Installation & Setup

### Prerequisites

* Node.js (v14 or higher)
* MongoDB (local or remote instance)
* An API key from Together AI (<https://together.xyz/>)

### Installation

1. **Clone the Repository**

bashCopy

git clone https://github.com/your-username/priyanshu6091-document\_qna.git

cd priyanshu6091-document\_qna

1. **Install Dependencies**

bashCopy

*# Backend*

cd backend

npm install

*# Frontend*

cd ../frontend

npm install

1. **Set Up Environment Variables**

Create a .env file in the backend directory and add the following:

envCopy

MONGO\_URI=mongodb://localhost:27017/docQA

API\_KEY=your-together-ai-api-key

1. **Start the Application**

bashCopy

*# Run Backend*

cd backend

npm start

*# Run Frontend*

cd ../frontend

npm run dev

**Note:** Ensure Node.js and MongoDB are installed.

## Usage

1. **Open the Application**

Open your browser and navigate to http://localhost:5173.

1. **Upload a Document**
   * Click on the "Choose File" button to select a PDF or text file.
   * Click the "Upload" button to upload the file.
2. **Ask a Question**
   * Once the file is uploaded, enter a question in the input field.
   * Click the "Ask" button to get an answer based on the document's content.

## API Endpoints

### Backend API

* **POST /upload**: Uploads a file and saves its content to the database.
  + Request Body: multipart/form-data with a single file field named file.
  + Response: { message: "File uploaded successfully", docId: "document\_id" }
* **POST /ask**: Generates an answer to a question based on the document's content.
  + Request Body: JSON object with docId and question.
  + Response: { answer: "AI-generated answer" }

## Contributing

Contributions are welcome! Please follow these steps:

1. **Fork** the repository.
2. Create a **new branch** (feature-branch).
3. Commit & push your changes.
4. Open a **Pull Request**.

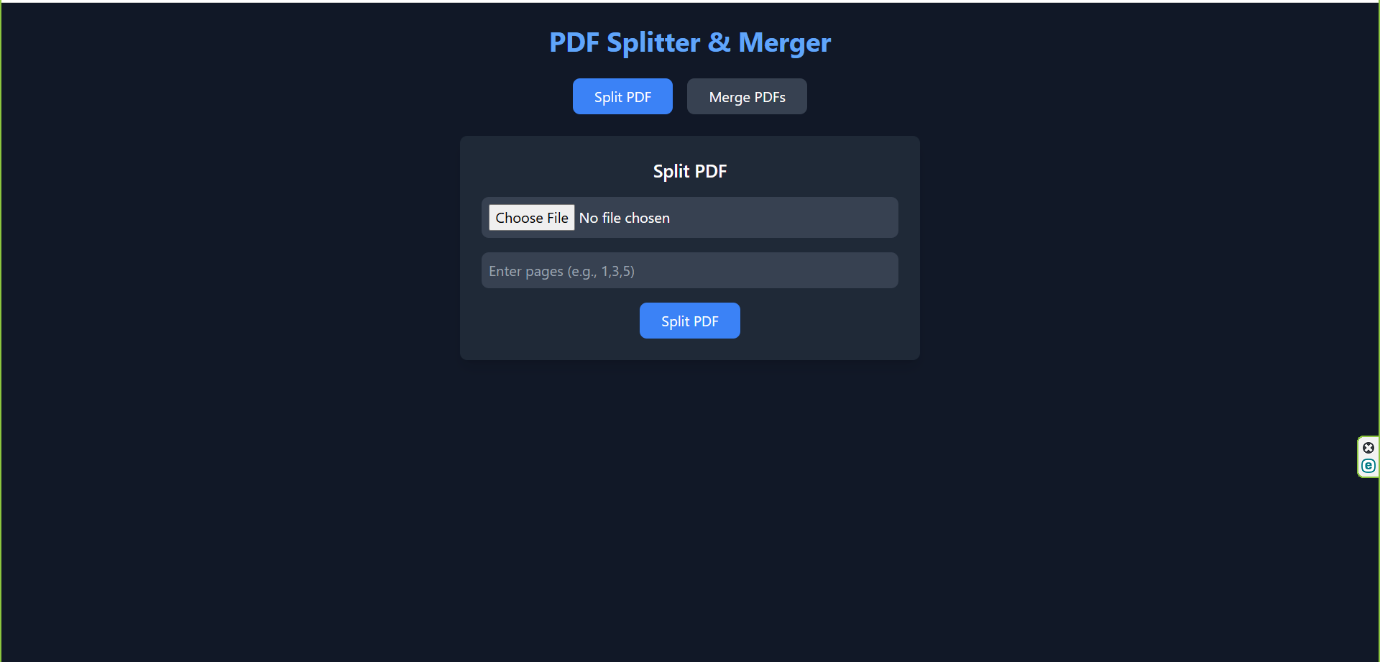
## License

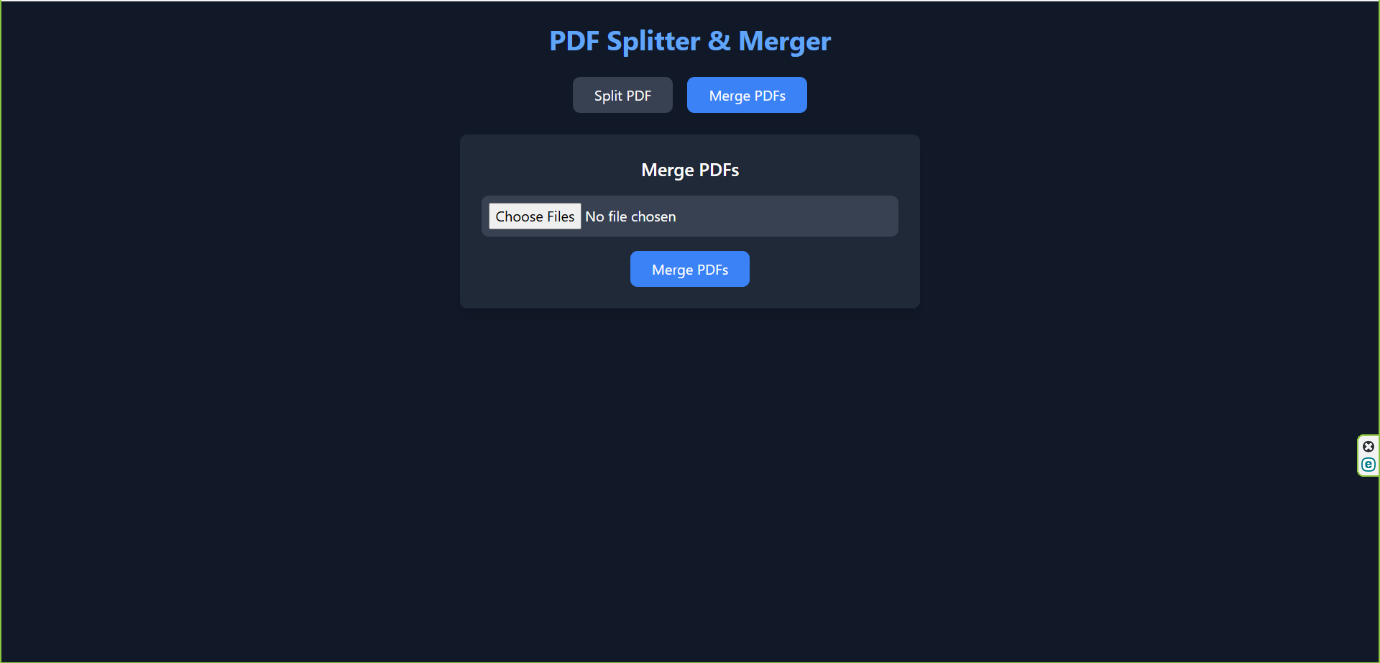
This project is licensed under the **MIT License**. Feel free to use and modify it.

# PDF Splitter & Merger Documentation

## Project Overview

The PDF Splitter & Merger is a full-stack web application that allows users to split a PDF into selected pages and merge multiple PDFs into a single file. It is built using the MERN stack (MongoDB, Express, React, Node.js) with pdf-lib for handling PDFs.





## Directory Structure

Copy

priyanshu6091-pdf-splitter/

├── README.md

├── backend/

│ ├── package-lock.json

│ ├── package.json

│ ├── server.js

│ ├── controllers/

│ │ └── pdfController.js

│ └── routes/

│ └── pdfRoutes.js

└── frontend/

└── pdf-tool/

├── README.md

├── eslint.config.js

├── index.html

├── package-lock.json

├── package.json

├── postcss.config.js

├── tailwind.config.js

├── vite.config.js

├── public/

└── src/

├── App.css

├── App.jsx

├── index.css

├── main.jsx

├── output.css

├── assets/

└── components/

├── Merger.jsx

├── Splitter.jsx

└── Upload.jsx

## Features

* **Upload PDF files**
* **Select specific pages to extract**
* **Merge multiple PDFs into one**
* **Real-time error handling & validation**
* **User-friendly, modern UI with Tailwind CSS**
* **Responsive Design for all devices**

## Tech Stack

| **Technology** | **Usage** |
| --- | --- |
| **Frontend** | React, Tailwind CSS, Vite, Axios |
| **Backend** | Node.js, Express.js, Multer |
| **PDF Handling** | pdf-lib |
| **Storage** | MongoDB (if needed for user history) |
| **Deployment** | Vercel (Frontend), Render (Backend) |

## Installation & Setup

### Prerequisites

* Node.js (v14 or higher)
* MongoDB (local or remote instance)

### Installation

1. **Clone the Repository**

bashCopy

git clone https://github.com/your-username/priyanshu6091-pdf-splitter.git

cd priyanshu6091-pdf-splitter

1. **Install Dependencies**

bashCopy

*# Backend*

cd backend

npm install

*# Frontend*

cd ../frontend/pdf-tool

npm install

1. **Start the Application**

bashCopy

*# Run Backend*

cd backend

npm start

*# Run Frontend*

cd ../frontend/pdf-tool

npm run dev

**Note:** Ensure Node.js and MongoDB are installed.

## Usage

1. **Open the Application**

Open your browser and navigate to http://localhost:5173.

1. **Split a PDF**
   * Click on the "Split PDF" button.
   * Upload a PDF file.
   * Enter the pages you want to extract (e.g., 1,3,5).
   * Click the "Split PDF" button to download the split PDF.
2. **Merge PDFs**
   * Click on the "Merge PDFs" button.
   * Upload multiple PDF files.
   * Click the "Merge PDFs" button to download the merged PDF.

## API Endpoints

### Backend API

* **POST /api/pdf/split**: Splits a PDF into selected pages.
  + Request Body: multipart/form-data with a single PDF file and selected pages (array).
  + Response: Downloadable split PDF file.
* **POST /api/pdf/merge**: Merges multiple PDFs into one.
  + Request Body: multipart/form-data with multiple PDF files.
  + Response: Merged PDF file for download.

## Contributing

Contributions are welcome! Please follow these steps:

1. **Fork** the repository.
2. Create a **new branch** (feature-branch).
3. Commit & push your changes.
4. Open a **Pull Request**.

## License

This project is licensed under the **MIT License**. Feel free to use and modify it.