**User Story:**

As a user, I want to be able to browse, upload, and download PDF files using a simple UI, so that I can manage documents efficiently.

**Acceptance Criteria:**

1. **Browse PDFs:**
   * Upon opening the UI, I should see a list/grid view displaying all available PDF documents.
   * Each document entry should display its name and possibly other relevant information (e.g., upload date).
   * If there are no PDF documents available, a message should be displayed indicating so.
2. **Upload PDF:**
   * There should be a clearly visible option/button to upload a PDF document.
   * Upon selecting the upload option, a file dialog should open, allowing the user to choose a PDF file from their local system.
   * After selecting a file, the UI should initiate an upload process to the backend RESTful API.
   * Upon successful upload, the UI should display a confirmation message and refresh the document list to reflect the newly uploaded document.
3. **Download PDF:**
   * Each document entry in the list/grid should have a clickable element (e.g., button or link) representing the download action.
   * Upon clicking the download element, the UI should trigger a request to the backend API to retrieve the PDF file associated with the clicked document.
   * The backend API should return the PDF file along with its UUID (document ID).
   * The downloaded file should retain its original format and content.
4. **Document Record Creation:**
   * When a PDF file is uploaded via the UI, the backend API should create a corresponding record in the Docs Table.
   * The record should include metadata such as document name, upload date, and a unique UUID for the document.
   * The UUID should be returned to the UI upon successful upload for reference in download requests.
5. **Error Handling:**
   * If there are any errors during upload or download processes (e.g., network issues, server errors), appropriate error messages should be displayed to the user.
   * Error messages should be clear and informative, guiding the user on how to proceed or resolve the issue.
6. **Security:**
   * The UI should validate uploaded files to ensure they are indeed PDF documents and not malicious files.
   * Access to the upload and download functionalities should be restricted based on user authentication and authorization mechanisms.
   * Communication between the UI and backend API should be secure, utilizing HTTPS protocol and possibly other security measures like authentication tokens.

By fulfilling these acceptance criteria, the user will have a functional UI for browsing, uploading, and downloading PDF files securely and efficiently, integrated with a backend RESTful API for document management.

Actions:

1. Create Technical design document
2. Write Code
3. Write Unit tests
4. Commit Code to Repository (github)
5. Deploy this service on any free VM/aws etc