**Project Name: Document Management System (DMS)**

**Table of Contents**

1. Features
2. Tech Stack
3. Architecture
4. API Documentation
5. **Features**

* **Document Upload**: Users can upload various types of documents, including PDFs, Word files, excel files and text files.
* **Document Search & Filter**: Search documents by title, type, tags, and more. Apply filters to refine the search results.
* **Pagination**: Paginate document results for efficient viewing and navigation.

**2. Tech Stack**

* **Backend**: Java, Spring Boot, Spring Data JPA
* **Database**: MySQL
* **Testing**: JUnit, Mockito
* **Build Tools**: Maven
* **Logging**: SLF4J

**3. Architecture**

**Overview**

The Document Management System follows a layered architecture consisting of the following components:

* **Controller Layer**: Manages HTTP requests and responses (REST API).
* **Service Layer**: Contains the business logic and interacts with the database via repositories.
* **Repository Layer**: Interfaces with the mySQL database through Spring Data JPA.

**5. API Documentation**

**Base URL**

http://localhost:8080/document

**Endpoints**

**1. GET /filter**

* **Description**: Retrieve all documents with pagination support based on author and type filter
* **Query Parameters**:
  + author (optional: document author
  + type (optional): document type
  + page (optional): The page number (default: 0)
  + size (optional): The number of documents per page (default: 10)
  + sort (optional): Sorting criteria in the format field,asc|desc (e.g., name,asc)
* **Response**:
  + 200 OK: List of documents

**2. POST /upload**

* **Description**: Upload a new document.
* **Request Body**: Multipart file upload
* **Response**:
  + 201 Created: Document successfully uploaded
  + 400 Bad Request: Invalid file format

**Base URL**

http://localhost:8080/qna

**Endpoints**

**3. GET /search**

* **Description**: Search for documents based on keyword(may include partial matches).
* **Resquest Parameter**: string keyword
* **Response**:
  + 200 OK: list of documents

**4. GET /searchExact**

* **Description**: Search for documents based on keyword.
* **Resquest Parameter**: string keyword
* **Response**:
  + 200 OK: list of documents