AISSMS
INSTITUTE OF INFORMATION TECHNOLOGY
ADDING VALUE TO ENGINEERING

# Department of Computer Engineering

A Mini Project on WebTechnology

## “Document Log”

SUBMITTED TO THE DEPARTMENT OF COMPUTER ENGINEERING

AISSMS IOIT

**T. Y. BTech Engineering**

**SUBMITTED BY**

| Student Roll No | Student Name |
|---|---|
| 95 | Sarthak Annasaheb Nage |

## Department of Computer Engineering

## CERTIFICATE

This is to certify that the project report

# “Document Log”

| Student Roll No | Student Name |
|---|---|
| **95** | **Sarthak Annasaheb Nage** |

Submitted by

is a bonafide student of this institute and the work has been carried out by him/her under the supervision of **prof. Farhana Sayeda** and it is approved for the partial fulfillment of the Department of Computer Engineering AISSMS IOIT.

( Guide**)** **(Dr. S. N. ZAWARE)**

Head,

Department of Computer Engineering Department of Computer
Engineering

Place : Pune Date :

# Index

# 1.ABSTRACT

## 1.1 CONTEXT

A. **Project Overview**: You are creating a **Document Log project** using Django. In this project:

- You can add, view, delete, and upload PDF/DOC files.
- Each document has associated details such as name, source/author, description, and date.
- The **home page** displays a list of all documents.
- A separate **Add Document** page allows users to input new document details and upload files.
- Users can delete both document entries and their uploaded files.

B. **Report Requirement**: You are working on a **project report** for this system, and I provided a detailed structure, including:

- Introduction, Objectives, and System Scope.
- System Requirements (hardware and software).
- System Design (database schema and functional modules).
- Implementation details (Django models, views, forms, and templates).
- Testing, Conclusion, and Future Scope.

## 1.2 PROBLEM STATEMENT

- The **Document Log System** seeks to provide a solution by offering a web-based platform for efficient document management, enabling users to upload, view, and delete documents .

## 1.3. OBJECTIVES

The main objectives of the Document Log System are:

1. **Centralized Document Management**
   Provide a centralized platform for users to store and organize their documents efficiently**.**
2. **Document Upload and Storage**
   Enable users to upload documents in PDF or DOC format and store them securely.
3. **Metadata Association**
   Allow users to associate each document with relevant metadata, including:
   - Document Name
   - Source/Author Name
   - Description
   - Date
4. **Document Retrieval and Display**
   Display a comprehensive list of uploaded documents on the home page, allowing users to view document details at a glance.
5. **Document Deletion**
   Facilitate the deletion of documents and their associated files, ensuring users can manage storage effectively.
6. **User-Friendly Interface**
   Provide an intuitive and easy-to-navigate interface for adding, viewing, and managing documents.
7. **Scalability and Extensibility**
   Build a system that can be easily extended to include additional features such as search functionality, user authentication, and advanced filtering in the future**.**

# 2.DESCRIPTION

## 2.1 User Role:

The **Document Log System** is a web-based application developed using Django that allows users to efficiently manage their documents. The system provides a streamlined solution for uploading, viewing, and organizing documents along with their associated metadata. It is designed to simplify document management tasks and ensure that important files are easily accessible.

## 2.2 Key Features:

1. **Document List on Home Page**
   The home page displays a list of all uploaded documents, showing essential details like the document name, source/author, description, and date. Users can quickly view the list and manage their documents.
2. **Add New Document**
   Users can add new documents through a dedicated form available on a separate page. This form includes fields for document metadata and file upload functionality for PDF/DOC formats.
3. **File Upload and Storage**
   Uploaded files are stored securely in the system. The system ensures that each document is associated with its respective file, making retrieval and management easier.
4. **Document Deletion**
   Users can delete both the document entry and the uploaded file, ensuring they maintain control over storage and document lifecycle.
5. **Dynamic and Scalable Design**
   The application is built to handle a growing number of documents efficiently. The system is also designed with scalability in mind, allowing for future enhancements such as user authentication, search, and categorization.

## 2.3Technical Aspects:

- **Backend**: Built using the Django framework, ensuring robust and secure management of data and files.
- **Frontend**: Uses HTML, CSS, and JavaScript for a user-friendly interface.
- **Database**: Utilizes SQLite for storing document metadata, with potential for migration to more scalable databases like PostgreSQL.
- **File Handling**: Uploaded files are stored in a dedicated directory and managed via Django's FileField.

This system aims to solve common document management challenges, offering an intuitive, efficient, and scalable solution.

## 3. SOFTWARE REQUIREMENTS:

### 1. Operating System

- Development: Compatible with Windows, Linux, or macOS.
- Deployment: Preferably Linux for server environments (e.g., Ubuntu).

### 2. Backend

- Django Framework (Version 4.x or higher)

  Used for developing the backend of the application, including database management, file handling, and business logic.

- Python (Version 3.8 or higher)

  Programming language required for Django development and running server-side scripts.

### 3. Frontend

- HTML5

  Used for structuring the web pages.

- CSS3

  Used for styling the web pages and ensuring a responsive design.

### 4. Database

- SQLite (Default Django database)

  Lightweight, embedded database for development and small-scale use.

- PostgreSQL or MySQL (Optional)

  Can be used for production environments requiring more robust data handling

### 5. File Storage

- Local File System

  Used to store uploaded PDF/DOC files during development.

- Cloud Storage (Optional)
  Services like AWS S3 or Google Cloud Storage can be used for scalable file storage in production.

## 6. Development Tools

- Code Editor/IDE:
  - Visual Studio Code
  - PyCharm (Optional)
- **Version Control:**
  - Git (Version control for managing code changes)
- **Package Manager:**
  - pip (Python package installer for installing dependencies)

## 7. Testing Tools

- Django Testing Framework
  For unit testing and integration testing of the application.
- Postman (Optional)
  For testing API endpoints if extended functionality involves RESTful APIs.

## 4. ALGORITHM:

### 1. Add a New Document

**Input: Document Name, Source/Author, Description, Date, and File (PDF/DOC).**
**Output: New document entry is added, and the file is uploaded.**

**Steps:**

1. Start.
2. Navigate to the Add Document page.
3. Fill in the required fields:
    - Document Name
    - Source/Author Name
    - Description
    - Date
4. Select the document file to upload.
5. Click Submit.
6. Validate input data:
    - Check for empty fields.
    - Verify file format (PDF/DOC).
7. If validation passes:
    - Save document details in the database.
    - Upload and store the file in the designated directory.
8. Redirect to the Home Page.
9. Display the updated list of documents.
10. End.

**2. View Document List**

**Input**: None.
**Output**: Display a list of documents with their details.

**Steps**:

1. Start.
2. Navigate to the **Home Page**.
3. Query the database for all document records.
4. Display document details in a table, including:
   - Document Name
   - Source/Author Name
   - Description
   - Date
   - File link (if uploaded).
5. End.

**3. Delete a Document**

**Input**: Document ID.
**Output**: Document and its associated file are deleted.

**Steps**:

1. Start.
2. On the **Home Page**, identify the document to delete.
3. Click the **Delete** button associated with the document.
4. Confirm the delete action.
5. Locate the document in the database using the provided ID.
6. Delete the record from the database.
7. If a file is associated:
   - Delete the file from the storage directory.
8. Refresh the **Home Page** to update the document list.

9. End.

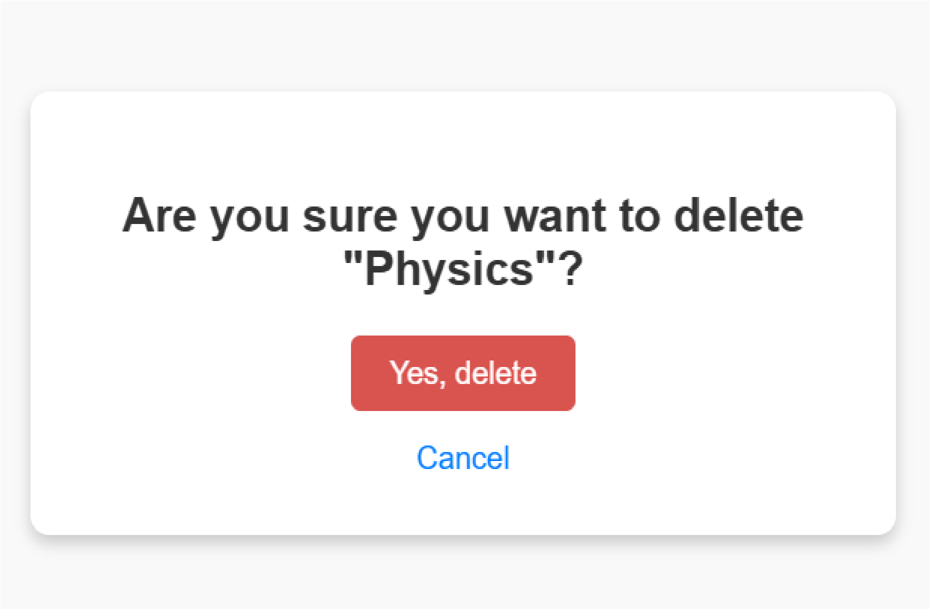
**4. Upload and Store Document File**

**Input**: File (PDF/DOC).
**Output**: File is stored in the system.

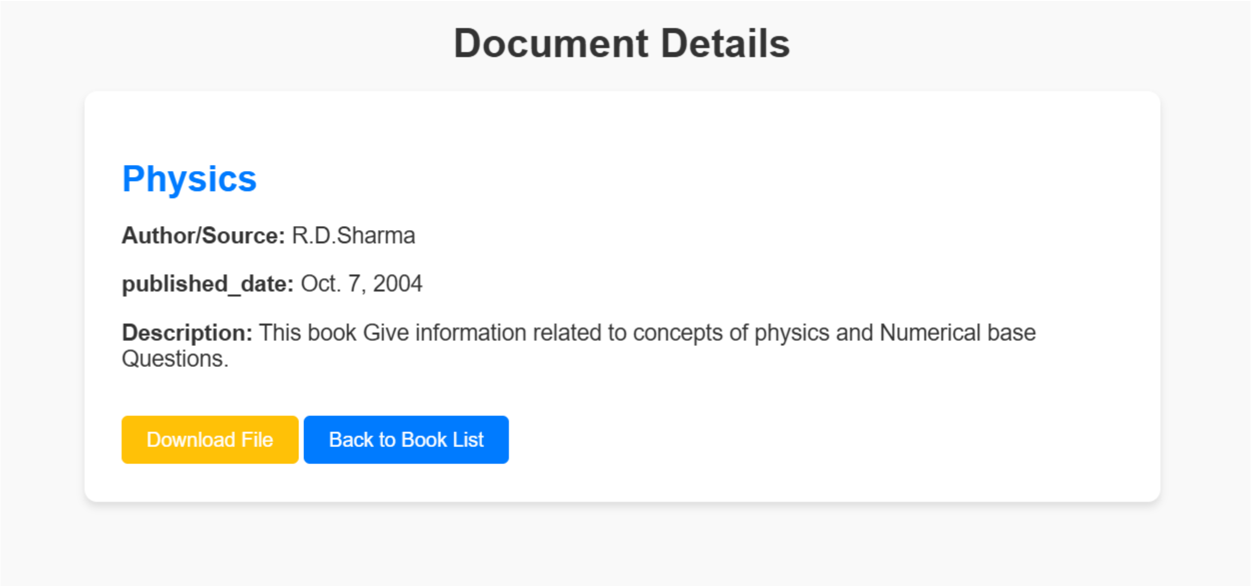
**Steps**:

1. Start.
2. Select a file for upload.
3. Verify the file type (allow only PDF/DOC).
4. Generate a unique file name to avoid conflicts.
5. Save the file in the designated directory (/documents/).
6. Update the file path in the document database record.
7. End.

# 6.SCREENSHOTS



Are you sure you want to delete "Physics"?



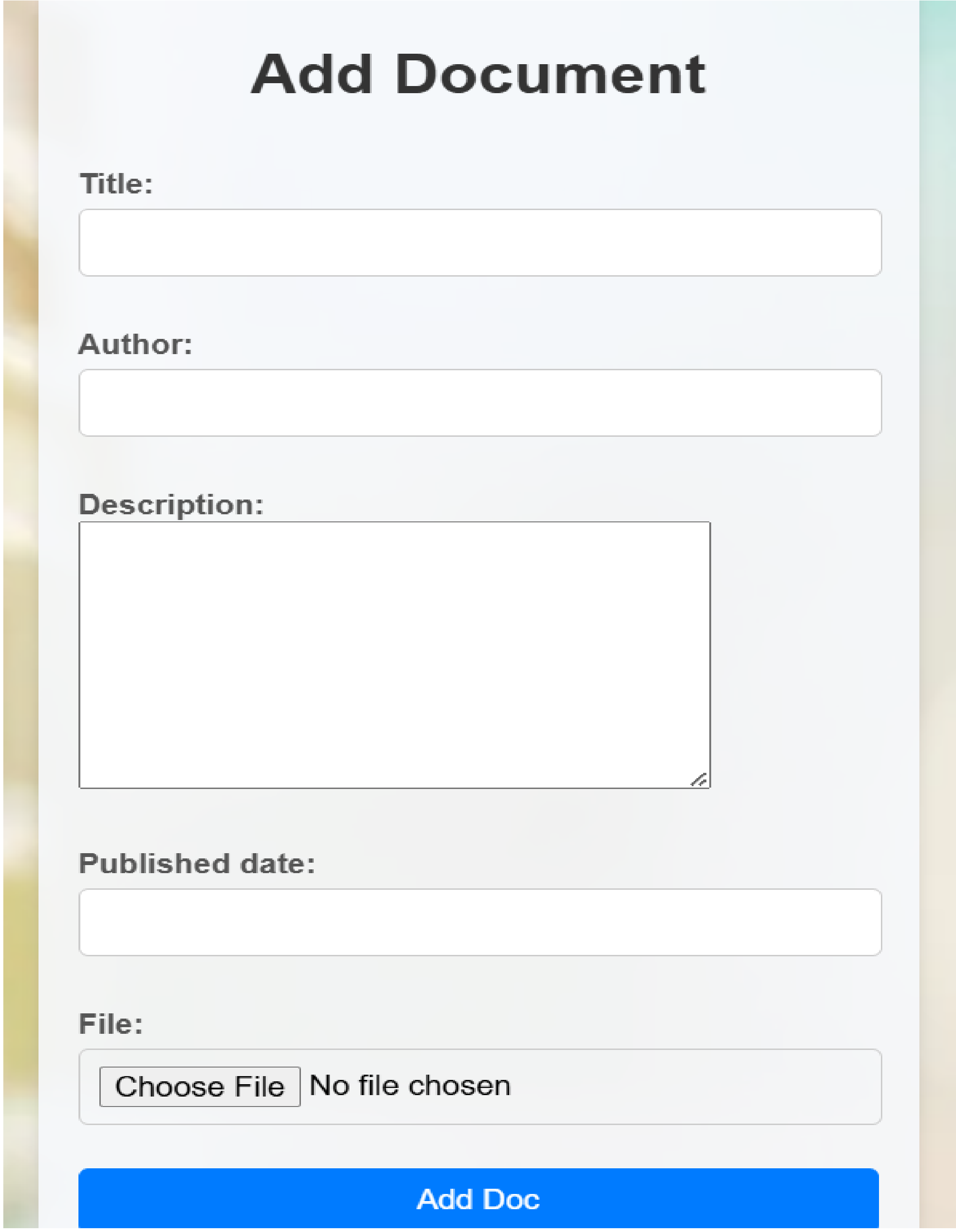
## Document Details

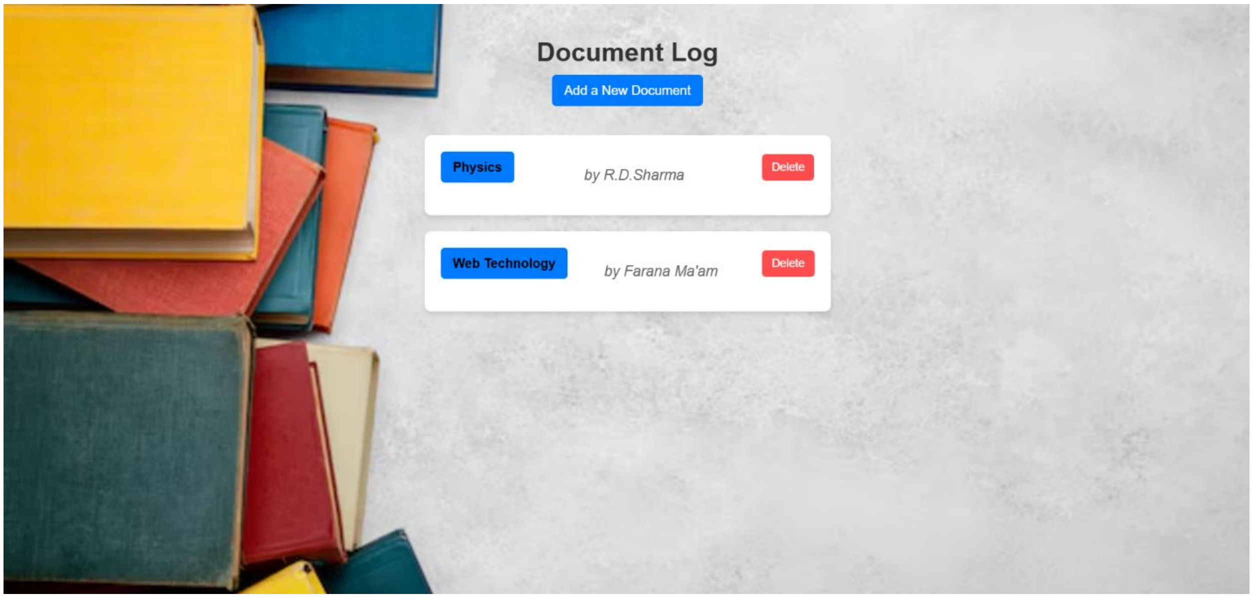
### Physics

**Author/Source:** R.D.Sharma

**published_date:** Oct. 7, 2004

**Description:** This book Give information related to concepts of physics and Numerical base Questions.


Add Document
Title:
Author:
Description:
Published date:
File:
Choose File
No file chosen
Add Doc



Document Log

Physics — by R.D.Sharma

Web Technology — by Farana Ma'am

## 7.ADVANTAGES AND DISADVANTAGES OF SYSTEM

### Advantages

1. **Centralized Document Management**
   - All documents are stored in one place, making it easy to manage and access them.
2. **Efficient File Handling**
   - Users can upload and store document files (PDF/DOC) securely, ensuring that important files are readily available.
3. **Metadata Organization**
   - Each document is associated with metadata (name, source/author, description, and date), improving organization and searchability.
4. **User-Friendly Interface**
   - The intuitive design allows users to quickly add, view, and delete documents without technical expertise.
5. **Data Integrity**
   - The system ensures the integrity of data by validating user inputs and file types before storing them.
6. **Scalability**
   - The system can be easily extended to include additional features like user authentication, advanced search, and categorization.
7. **Improved Productivity**
   - Simplifies document management, reducing time spent searching for and organizing files.
8. **Cost-Effective**
   - Being developed using open-source technologies like Django and SQLite reduces the overall cost of development and deployment.

### Disadvantages

1. **Limited to Basic Features**
   - The current version lacks advanced functionalities like document version control, categorization, and detailed search filters.
2. **Storage Dependency**
   - Uploaded files are stored locally, which could lead to storage limitations as the number of documents grows. Integration with cloud storage is recommended for scalability.
3. **No User Authentication**

   - The system doesn’t include a user login mechanism, which could lead to unauthorized access in multi-user environments.

4. **Manual Metadata Entry**
   - Users must manually input metadata (document name, source, etc.), which could lead to human errors.

5. **Limited Search Functionality**
   - Currently, there is no advanced search or filtering feature to quickly find specific documents based on metadata.

# 8.APPLICATIONS

**1.Corporate Document Management**

- Store and manage internal documents such as reports, meeting minutes, and policies.
- Organize documents by associating metadata for quick retrieval.
- Facilitate controlled deletion of outdated documents.

**2. Educational Institutions**

- Manage academic records, research papers, assignments, and course materials.
- Provide a centralized repository for faculty and students to upload and access important documents.

**3. Legal Firms**

- Manage case files, legal agreements, and court documents efficiently.
- Provide quick access to relevant case details through metadata association.

**4. Research and Development (R&D)**

- Store research papers, experimental data, and project reports.
- Organize and access documents related to ongoing or past research work easily.

**5. Personal Document Management**

- Individuals can use the system to store and manage personal documents such as resumes, certificates, and legal paperwork.

## 9.Conclusion

The **Document Log System** provides an efficient and user-friendly solution for managing documents by allowing users to upload, store, view, and delete files, along with associating relevant metadata such as document name, source/author, description, and date. Built with Django, the system ensures secure storage and easy retrieval of documents, making it ideal for various applications like corporate document management, education, healthcare, and legal sectors. The intuitive interface simplifies the document management process, enhancing productivity and organization. While it currently offers basic functionalities, such as file handling and metadata entry, the system is scalable, with potential for future enhancements like user authentication, advanced search, and cloud storage integration. Overall, the **Document Log System** addresses the common challenges in document management, offering a streamlined and effective solution suitable for both individuals and organizations.

## 10.References

1.Django Software Foundation. (n.d.). **Django Documentation**. Retrieved from https://www.djangoproject.com/

- Official documentation for the Django web framework, which provides detailed guidance on setting up and developing web applications.

2.W3Schools. (n.d.). **HTML, CSS, and JavaScript Tutorials**. Retrieved from https://www.w3schools.com/

- A comprehensive online resource for learning HTML, CSS, and JavaScript, used in the frontend development of web applications.

3.PostgreSQL Global Development Group. (n.d.). **PostgreSQL Documentation**. Retrieved from https://www.postgresql.org/docs/

- Official documentation for PostgreSQL, a powerful open-source relational database management system, useful for scaling the Document Log System.

4.Python Software Foundation. (n.d.). **Python Documentation**. Retrieved from https://docs.python.org/3/

- Official documentation for Python, the programming language used for backend development with Django.