DIGITAL LIBRARY MANAGEMENT WITH CLOUD STORAGE

PROJECT REPORT 2025-2026

Submitted By

24500420 KRISHNAVENI S

24500415 AJAY P

24500425 RAMYA R

24500429 SATHISHKUMAR S

Under the guidance of

Mrs.G.SUGANTHI B.E (CSE).,

LECT/COMPUTER

Diploma in Information Technology

of the directorate of Technical Education, Government of Tamil Nadu

DEPARTMENT OF INFORMATION TECHNOLOGY

AKT MEMORIAL POLYTECHNIC COLLEGE

KALLAKURICHI-606202.

AKT MEMORIAL POLYTECHNIC COLLEGE

KALLAKURICHI-606202

Department of Information Technology

BONAFIDE CERTIFICATE

This is certified that this project work entitled **Digital Library Management with Cloud Storage** has been submitted **KRISHNAVENI S, AJAY P, RAMYA R, SATHISHKUMAR S** in the partial fulfilment of the requirements for the award of Diploma in Information Technology during the academic year 2025-2026, who carried out the project work under our supervision.

Project Guide	**Head of the Department**
Mrs.G.SUGANTHI B.E (CSE).,	**Mrs. C.SRIDEVI B.E.,(CSE)**
LECT/COMPUTER	**HOD – COMPUTER & IT**

This is to certify that_____ was examined for the project work viva-voce held on_____

INTERNAL EXAMINER EXTERNAL EXAMINER

Acknowledgement

We express our sincere and profound thanks to our chairman **Mr.A.K.T.Mahendiran B.Com.,** for creating this magnificent edifice of learning by providing all necessary facilities to complete diploma engineering course successfully.

We express our gratitude to our principal **Mr.P.K.Kabilar M.E.,MBA.,** for constant encouragement and facilities provided towards the successful completion of our project work.

At the same time we would like to express our heartfelt thanks to our head of the department **Mrs.C.Sridevi.,B.E.,(CSE).,** and also our project guide **Mrs.G.Suganthi B.E (CSE).,** for guiding and needful advices and time to complete this project.

We would like to show our gratitude to our department staffs for all instructions and guidance given throughout.

Our sincere thanks and affection to our parents and friends who gave hand in all our steps.

CONTENTS

ABSTRACT

The **Digital Library Management System with Cloud Storage** is a modern web-based application developed using **Laravel 12 framework** and deployed on a **Virtual Private Server (VPS)** environment. The system is designed to efficiently manage, store, and provide secure access to digital library resources such as e-books, research papers, journals, and multimedia content.

This project addresses the limitations of traditional library systems by introducing a **cloud-integrated platform** that allows users to access resources anytime and from anywhere. The system provides features such as **user authentication, role-based access control (admin, librarian, student), digital content upload, cloud storage integration, advanced search functionality, and real-time availability tracking**.

The application leverages cloud storage services to ensure **scalability, data redundancy, and secure file management**, reducing dependency on local storage infrastructure. Administrators can manage users, categorize resources, monitor usage, and generate reports, while users can browse, search, download, and bookmark digital materials.

Additionally, the system enhances user experience through a **responsive interface**, efficient indexing, and quick retrieval of information. Security measures such as **data encryption, secure login, and controlled access permissions** are implemented to protect sensitive data.

By integrating Laravel's robust backend capabilities with cloud storage solutions and VPS hosting, the system delivers a **high-performance, scalable, and reliable digital library solution** suitable for educational institutions and organizations.

This project demonstrates how modern web technologies and cloud computing can be effectively utilized to transform traditional libraries into **smart, digital knowledge management systems**.

1. INTRODUCTION

In the digital era, the rapid growth of information and the increasing demand for remote access to knowledge have transformed the way libraries operate. Traditional library systems, which rely heavily on physical resources and manual management, often face challenges such as limited accessibility, inefficient record handling, space constraints, and difficulty in tracking resource usage. To overcome these limitations, there is a need for a modern, scalable, and efficient solution that leverages current technologies.

The Digital Library Management System with Cloud Storage is designed to address these challenges by providing a centralized, web-based platform for managing and accessing digital resources. Developed using the Laravel 12 framework, this system ensures a robust and secure backend, while deployment on a Virtual Private Server (VPS) guarantees high availability and performance.

This system enables users such as students, faculty, and administrators to access digital resources including e-books, journals, research papers, and multimedia content from anywhere at any time. By integrating cloud storage services, the system eliminates dependency on local storage and provides benefits such as scalability, data backup, and secure file handling.

The application incorporates advanced features such as user authentication, role-based access control, intelligent search functionality, and real-time resource management. Librarians and administrators can efficiently manage digital assets, categorize content, monitor user activity, and generate reports. Users, on the other hand, can easily search, view, download, and bookmark resources through a user-friendly interface.

Moreover, the system emphasizes data security and integrity by implementing secure login mechanisms, encrypted data handling, and controlled access permissions. The use of modern web technologies ensures that the platform is responsive, fast, and accessible across multiple devices.

This project aims to demonstrate how integrating web development frameworks, cloud computing, and VPS hosting can modernize traditional library systems into intelligent digital platforms, thereby improving accessibility, efficiency, and overall user experience.

2. LITERATURE SURVEY

The evolution of library systems from manual record-keeping to computerized management has been a major area of study in information systems and academic administration. Earlier library management systems were primarily designed to automate routine activities such as book cataloging, member registration, issue and return processing, and fine calculation. These systems significantly reduced manual effort, improved accuracy, and helped libraries manage large volumes of data more efficiently. However, most early solutions were limited to local environments and focused mainly on physical books rather than digital resources.

With the rise of the internet and web technologies, web-based library management systems emerged as a more flexible alternative. Researchers and developers proposed online systems that allowed users to search catalogs, reserve books, and access library information remotely. These systems improved user convenience and administrative control, but many of them still depended on centralized local servers and lacked efficient support for large-scale digital content such as e-books, scanned documents, journals, and multimedia files.

The concept of **digital libraries** introduced a significant shift in library services. Digital libraries are designed not only to maintain bibliographic information but also to store and deliver complete digital content. Studies in this area emphasized the importance of metadata management, indexing, search optimization, and user-friendly retrieval mechanisms. Digital libraries have proven useful in educational institutions, research centers, and public knowledge repositories, where users need continuous access to learning materials. Despite these advantages, many digital library platforms face challenges related to storage limitations, access speed, backup reliability, and system scalability.

The advancement of **cloud computing** has provided a strong solution to these limitations. Cloud-based storage and service models have been widely discussed in recent literature as an effective way to manage large digital repositories. By using cloud infrastructure, digital library systems can achieve better scalability, on-demand storage expansion, data redundancy, and high availability. Researchers have pointed out that cloud-enabled systems reduce infrastructure costs while improving access to information from multiple devices and locations. Cloud storage also supports automated backup and disaster recovery, which are essential for protecting valuable academic resources.

Another important area highlighted in previous studies is **security and access control**. Since digital library systems store user information as well as copyrighted academic content, researchers have focused on authentication mechanisms, role-based authorization, secure file access, and encryption methods. Modern systems are expected to support multiple user roles such as admin, librarian, faculty, and student, each with controlled permissions. This has made security a central requirement in the design of web-based digital library platforms.

Recent literature also emphasizes the role of modern web frameworks in building scalable and maintainable systems. Frameworks such as Laravel have gained attention because they support MVC architecture, routing, authentication, database abstraction, and built-in security features. These capabilities make Laravel a suitable platform for developing advanced academic management applications, including digital libraries. In addition, hosting such systems on **VPS servers** offers better control, dedicated resources, and deployment flexibility compared to shared hosting environments.

From the review of existing studies and systems, it is clear that although many library solutions are available, there is still a need for an integrated platform that combines **digital resource management, cloud storage, secure user access, efficient search, and reliable VPS deployment** in a single system. The proposed **Digital Library Management with Cloud Storage** project is developed to fulfill this need by bringing together modern web development practices and cloud-based file management into one unified solution.

Key Findings from Literature Review

1. Traditional library systems are effective for physical book tracking but are not suitable for modern digital resource access.
2. Web-based library systems improved accessibility, but many lack scalability and strong digital content support.
3. Digital libraries enable remote access to educational resources, but storage and maintenance remain major challenges.
4. Cloud storage provides scalability, backup, reliability, and easier access management for digital assets.
5. Secure authentication and role-based access are essential in academic digital platforms.
6. Laravel framework and VPS hosting provide a strong technical foundation for building a secure and scalable digital library management system.

3. PROPOSED SYSTEM

The proposed system, **Digital Library Management with Cloud Storage**, is a web-based application developed using **Laravel 12** and deployed on a **Virtual Private Server (VPS)** to provide a scalable, secure, and high-performance environment. The system is designed to efficiently manage digital resources and provide seamless access to users through cloud integration.

Overview of the Proposed System

The system replaces traditional library operations with a fully digital platform where all resources such as **e-books, research papers, journals, and multimedia files** are stored in cloud storage and accessed through a centralized web application. It enables users to access learning materials anytime and from anywhere, ensuring flexibility and convenience.

The system supports multiple users with different roles, including **Administrator, Librarian, and Student/User**, each having specific permissions and functionalities. The integration of cloud storage ensures that large volumes of data can be stored, managed, and retrieved efficiently without overloading the local server.

Key Features of the Proposed System

1. **User Authentication and Role Management**
   Secure login system with role-based access control. Admins manage the system, librarians manage resources, and users access content.
2. **Digital Resource Management**
   Allows uploading, organizing, categorizing, and managing digital files such as PDFs, e-books, and videos.
3. **Cloud Storage Integration**
   All files are stored in cloud platforms (AWS S3 / Google Cloud / DigitalOcean Spaces), ensuring scalability, backup, and high availability.

4. **Advanced Search System**

   Users can search resources using keywords, categories, authors, or titles with fast and accurate results.
5. **Online Access and Download**

   Users can view or download digital content securely based on permissions.
6. **Bookmark and Favorites**

   Users can save resources for quick access later.
7. **Admin Dashboard**

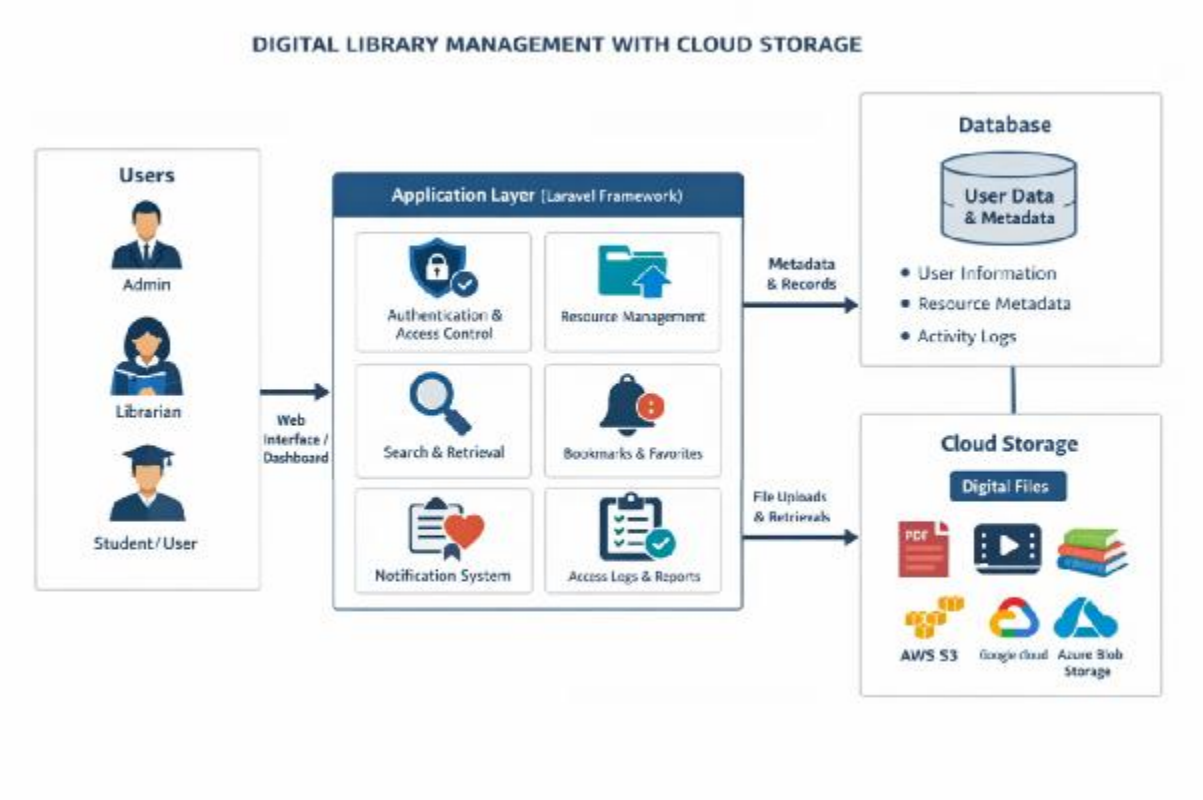
   Provides complete control over users, resources, categories, and system activities with analytics and reports.
8. **Activity Monitoring and Logs**

   Tracks user actions such as downloads, uploads, and login history.
9. **Notifications and Alerts**

   Sends notifications for new uploads, updates, or important announcements.
10. **Responsive UI Design**

    Accessible across desktops, tablets, and mobile devices

3.1 ARCHITECTURE



The **Digital Library Management with Cloud Storage** system follows a **cloud-enabled multi-tier architecture** designed to manage digital resources efficiently, provide secure access to users, and ensure reliable storage of library content. This architecture focuses on the interaction between the **user interface**, **application logic**, **database**, and **cloud storage services**, without depending on server deployment details.

1. Architecture Overview

The architecture is divided into the following major layers:

1. Presentation Layer

This is the front-end layer through which users interact with the system. It includes:

- Login page
- User dashboard

- Admin dashboard
- Resource upload page
- Search and filter interface
- Category and author browsing pages
- Bookmarks and download history pages

This layer provides a user-friendly and responsive interface for students, faculty, librarians, and administrators.

2. Application Layer

This is the core processing layer of the system. It handles:

- User authentication and authorization
- Resource upload and validation
- Search processing
- Metadata management
- Access control
- Notifications
- Download tracking
- Bookmark handling

The application layer connects the user interface with the database and cloud storage, ensuring smooth execution of all library functions.

3. Database Layer

This layer stores all structured information required for system operation, including:

- User details
- Roles and permissions
- Resource metadata
- Categories and authors
- Upload records
- Bookmark data
- Download history
- Activity logs
- Notifications

The database stores only the resource details and file references, not the actual large digital files.

4. Cloud Storage Layer

This is the most important part of the architecture. It stores all digital files such as:

- E-books
- PDF documents
- Journals
- Research papers
- Thesis files
- Audio/video learning materials
- Images and scanned documents

The cloud storage service provides:

- Scalable file storage
- Secure access
- Backup and redundancy
- Faster retrieval
- Reduced local storage usage

Instead of storing files directly in the application database, the system saves them in cloud storage and keeps only the file URL or storage path in the database.

2. Main Architectural Components

a) User Interface Module

This module allows users to interact with the digital library system. Different users access different features:

- **Admin** manages users, categories, reports, and overall system settings
- **Librarian** uploads and manages resources
- **Student/User** searches, views, downloads, and bookmarks resources

b) Authentication and Access Control Module

This component verifies login credentials and controls access based on user roles. It ensures that:

- Only authorized users can upload or manage resources

- Admin functions are protected
- Users can only access permitted content

c) Resource Management Module

This module handles:

- Adding new digital resources
- Editing metadata
- Organizing resources by category, author, and subject
- Updating file information
- Deleting outdated resources

d) Search and Retrieval Module

This component allows users to search digital content using:

- Title
- Author
- Subject
- Category
- Keywords
- Publication year

It improves user experience by enabling quick and accurate retrieval of resources.

e) Cloud File Upload Module

When a librarian or admin uploads a file:

1. The file is validated
2. The file is transferred to cloud storage
3. The storage path or public/private URL is generated
4. The file metadata is stored in the database

This ensures efficient file handling and avoids overloading the main system.

f) Download and Access Tracking Module

This module records:

- Which user accessed which file
- Download date and time
- Number of downloads
- Frequently accessed resources

This information helps in generating reports and understanding user behavior.

g) Bookmark and Favorite Module

Users can save preferred resources for future use. This module improves usability by allowing quick access to important materials.

h) Notification Module

This module sends alerts related to:

- Newly uploaded resources
- Updated content
- Important library announcements
- Account-related notifications

3.2 MODULES DESCRIPTION

1. User Registration and Authentication Module

This module manages account creation, login, logout, password reset, and secure user verification. It ensures that only authorized users can access the system. Users such as Admin, Librarian, and Student register or are created by the administrator. Authentication is handled securely using encrypted passwords and session management. This module is important because it acts as the entry point of the system and protects digital resources from unauthorized access.

Functions:

- User registration
- Secure login and logout
- Password reset
- Email verification
- Session handling

2. Role-Based Access Control Module

This module controls what each type of user can access and perform in the system. The Admin has full access to all modules, the Librarian manages resources and users, and Students/Users can search, view, and download content. This module improves security and system organization by ensuring that features are accessed only by appropriate roles.

Functions:

- Admin privileges management
- Librarian permissions
- Student/User restricted access
- Role assignment

- Permission control

3. Digital Resource Upload and Management Module

This is one of the core modules of the project. It allows Admin or Librarian to upload digital resources such as PDFs, e-books, journals, thesis documents, scanned notes, and multimedia files. The system collects metadata like title, author, category, publication year, keywords, and file description. It also supports editing, updating, and deleting resources.

Functions:

- File upload
- Metadata entry
- Edit resource details
- Delete outdated files
- Organize resources systematically

4. Cloud Storage Integration Module

This module handles storage of uploaded digital resources in cloud storage platforms instead of storing them directly on the local server. It ensures better scalability, backup, and file availability. The module generates and manages file paths, access URLs, and storage references, while the actual metadata is stored in the database.

Functions:

- Upload files to cloud storage
- Generate secure file URLs
- Store file references in database
- Retrieve files from cloud storage
- Manage storage usage

5. Cataloging and Classification Module

This module organizes resources in a structured format so that users can easily browse and locate materials. Resources are categorized by subject, department, course, author, publication type, and tags. Proper classification improves search efficiency and overall system usability.

Functions:

- Category creation
- Author management
- Subject classification
- Tagging and indexing
- Organized digital catalog

6. Search and Retrieval Module

This module allows users to search for digital resources quickly and accurately. Users can search using title, author name, category, subject, keyword, or publication year. It improves the user experience by helping them find relevant materials in a short time.

Functions:

- Keyword search
- Advanced filters
- Search by title, author, category
- Fast retrieval of metadata
- Direct access to files

7. Download and Access Tracking Module

This module records every important user action related to digital resources. It stores details about which user viewed or downloaded which file, along with date and time. These records help the administration understand user activity and maintain system transparency.

Functions:

- Download history
- File access logs
- User activity tracking

- Usage statistics
- Report support

8. Bookmark and Favorite Module

This module allows users to save resources for later use. It improves convenience for students and researchers who frequently refer to certain materials. Users can maintain a personalized collection of favorite documents and quickly access them whenever needed.

Functions:

- Add bookmarks
- Remove bookmarks
- View saved resources
- Personalized reading list
- Quick future access

9. Notification and Announcement Module

This module is used to inform users about important events and updates in the system. Notifications may include newly uploaded books, updated materials, due announcements, or library messages. It improves communication between administrators and users.

Functions:

- New resource alerts
- Announcement display
- User notifications
- Update messages
- Library information broadcast

10. Reports and Analytics Module

This module generates useful reports for administrators and librarians. It provides details such as number of uploaded resources, most downloaded files, active users, category-wise resource counts, and overall usage trends. This module helps management make decisions based on actual system data.

Functions:

- Resource usage reports
- User activity reports
- Download statistics
- Category-wise reports
- System performance summary

4. IMPLEMENTATION

The implementation of the **Digital Library Management with Cloud Storage** system focuses on developing a secure, scalable, and user-friendly web application that manages digital learning resources efficiently. The system is implemented using a modern development approach where the frontend handles user interaction, the backend processes business logic, the database stores structured data, and cloud storage manages digital files such as e-books, journals, PDFs, and multimedia content.

The project is developed as a web-based application so that users can access the digital library through a browser without installing additional software. The implementation is divided into several stages including requirement analysis, system design, database creation, module development, cloud integration, testing, and deployment.

1. Requirement Analysis and Planning

In the first stage, the project requirements are identified based on the needs of a digital library. The major requirements include user management, secure login, digital resource upload, metadata storage, search functionality, cloud file storage, bookmarking, notifications, and reporting. User roles such as **Admin**, **Librarian**, and **Student/User** are defined clearly so that the access level of each user can be controlled during implementation.

At this stage, the system flow is planned, database tables are identified, and the interaction between modules is designed.

2. Frontend Implementation

The frontend is implemented to provide a responsive and easy-to-use interface for all users. Separate dashboards and pages are created for different operations such as:

- Login and registration

- Admin dashboard
- Librarian dashboard
- Resource upload form
- Resource listing page
- Search page
- Bookmark page
- Reports page
- Notification panel

The frontend includes forms for entering book details, uploading digital resources, managing categories, and searching records. The interface is designed to be simple and clean so that users can easily navigate through the system.

3. Backend Implementation

The backend handles the core business logic of the application. It processes requests from users, validates inputs, communicates with the database, and manages integration with cloud storage.

The backend implementation includes:

- User authentication and authorization
- Session management
- Role-based permission handling
- Resource upload processing
- Metadata insertion and update
- Search and filtering logic
- Bookmark management
- Notification generation
- Report generation
- Activity logging

When a user performs an action such as login, upload, search, or download, the backend processes that request and returns the required result to the frontend.

4. Database Implementation

The database is implemented to store all structured data required for system operation. Instead of storing large digital files directly in the database, only metadata and file reference paths are stored.

The database contains tables such as:

- users
- roles
- resources
- categories
- authors
- bookmarks
- downloads
- notifications
- activity_logs
- cloud_files

Each table is related properly using primary keys and foreign keys. This relational design ensures data consistency and makes the system easy to manage.

For example:

- The **users** table stores login and profile details
- The **resources** table stores book or file metadata
- The **categories** table stores subject-wise classification
- The **downloads** table stores download history
- The **bookmarks** table stores user favorite resources

5. Cloud Storage Implementation

One of the important parts of the system is cloud storage implementation. When a librarian or admin uploads a file, the file is not stored permanently in the local application folder. Instead, it is transferred to cloud storage.

The implementation flow is:

1. The user selects a digital file from the upload form
2. The system validates file type and size
3. The file is uploaded to cloud storage
4. A unique file path or URL is generated

5. The file reference is saved in the database along with metadata
6. When a user requests the file, the system retrieves it through the stored cloud path

This implementation reduces local storage dependency and improves scalability. It also makes backup and file access easier.

4.1 SOFTWARE ENVIRONMENT

1. Operating System

- Windows 10 / 11
- Linux (Ubuntu / CentOS)
- macOS

2. Frontend Technologies

- HTML5
- CSS3
- Bootstrap / Tailwind CSS
- JavaScript
- jQuery / AJAX

3. Backend Technologies

- PHP 8.x
- Laravel 12 Framework
- Composer

4. Database System

- MySQL / MariaDB
- phpMyAdmin

5. Cloud Storage

- AWS S3
- Google Cloud Storage
- DigitalOcean Spaces

4.2 SYSTEM REQUIREMENTS

1. Hardware Requirements

Development System Requirements

These are the minimum hardware requirements for developing and testing the project:

- **Processor:** Intel Core i3 / i5 or higher
- **RAM:** 8 GB minimum
- **Hard Disk:** 256 GB SSD or higher
- **Monitor:** 14-inch or above
- **Keyboard and Mouse:** Standard input devices
- **Internet Connection:** Stable broadband connection

These specifications are sufficient for coding, database handling, local server testing, and UI development.

Server Requirements (VPS Hosting)

For live deployment, the application requires a **Virtual Private Server (VPS)** with the following configuration:

Component	Specification
Server Type	Virtual Private Server (VPS)
CPU	8 Core Processor
RAM	32 GB
Storage	300 GB NVMe SSD

Component	Specification
Bandwidth	High-speed / Unlimited preferred
Operating System	Ubuntu / CentOS / AlmaLinux
Web Server	Apache or Nginx
Database Server	MariaDB
Control Panel	WHM / cPanel
Backup Support	Daily / Weekly Backup Recommended
SSL Certificate	Required for secure access

4.3 SAMPLE CODING

```
<!doctype html>
<html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1">

        <title>Laravel</title>

        <!-- Fonts -->
        <link href="https://fonts.googleapis.com/css?family=Nunito:200,600" rel="stylesheet">

        <!-- Styles -->
        <style>
            html, body {
                background-color: #fff;
                color: #636b6f;
                font-family: 'Nunito', sans-serif;
                font-weight: 200;
                height: 100vh;
```

```
    margin: 0;
}

.full-height {
    height: 100vh;
}

.flex-center {
    align-items: center;
    display: flex;
    justify-content: center;
}

.position-ref {
    position: relative;
}

.top-right {
    position: absolute;
    right: 10px;
    top: 18px;
}

.content {
    text-align: center;
}

.title {
    font-size: 84px;
}
```

```
        .links > a {
            color: #636b6f;
            padding: 0 25px;
            font-size: 13px;
            font-weight: 600;
            letter-spacing: .1rem;
            text-decoration: none;
            text-transform: uppercase;
        }

        .m-b-md {
            margin-bottom: 30px;
        }
    </style>
</head>
<body>
    <div class="flex-center position-ref full-height">
        @if (Route::has('login'))
            <div class="top-right links">
                @auth
                    <a href="{{ url('/home') }}">Home</a>
                @else
                    <a href="{{ route('login') }}">Login</a>

                    @if (Route::has('register'))
                        <a href="{{ route('register') }}">Register</a>
                    @endif
                @endauth
            </div>
        @endif
```

```
            <div class="content">
                <div class="title m-b-md">
                    Laravel
                </div>

                <div class="links">
                    <a href="https://laravel.com/docs">Docs</a>
                    <a href="https://laracasts.com">Laracasts</a>
                    <a href="https://laravel-news.com">News</a>
                    <a href="https://blog.laravel.com">Blog</a>
                    <a href="https://nova.laravel.com">Nova</a>
                    <a href="https://forge.laravel.com">Forge</a>
                    <a href="https://github.com/laravel/laravel">GitHub</a>
                </div>
            </div>
        </div>
    </body>
</html>
```

E- Book

```
@extends('admin::layout')

@component('admin::include.page.header')
    @slot('title', clean(trans('ebook::ebooks.ebooks')))

    <li class="nav-item">{{ clean(trans('ebook::ebooks.ebooks')) }}</li>
@endcomponent

@component('admin::include.page.table')
    @slot('title', clean(trans('ebook::ebooks.ebooks')))
```

```
    @slot('buttons', ['create'])
    @slot('resource', 'ebooks')
    @slot('name', clean(trans('ebook::ebooks.ebooks')))

    @slot('thead')
        <tr>
            @include('admin::include.table.select-
all',["name"=>trans('ebook::ebooks.ebook')])
            <th>{{ clean(trans('ebook::ebooks.table.bookcover')) }}</th>
            <th>{{ clean(trans('ebook::ebooks.table.title')) }}</th>
            <th>{{ clean(trans('ebook::ebooks.table.user')) }}</th>
            <th>{{ clean(trans('ebook::ebooks.table.password')) }}</th>
            <th>{{ clean(trans('ebook::ebooks.table.is_featured')) }}</th>
            <th>{{ clean(trans('ebook::ebooks.table.is_private')) }}</th>
            <th>{{ clean(trans('ebook::ebooks.table.views')) }}</th>
            <th>{{ clean(trans('admin::admin.table.status')) }}</th>
            <th data-sort>{{ clean(trans('admin::admin.table.created')) }}</th>
        </tr>
    @endslot
@endcomponent

@push('scripts')
    <script>
        new DataTable('#ebooks-table .table', {
            columns: [
                { data: 'checkbox', orderable: false, searchable: false, width: '3%' },
                { data: 'bookcover', orderable: false, searchable: false, width: '10%' },
                { data: 'title', name: 'translations.title', orderable: false, defaultContent: '' },
                { data: 'user.full_name', name: 'user.first_name', orderable: false,
defaultContent: '' },
                { data: 'password', name: 'password', orderable: false, defaultContent: '' },
```

```
            { data: 'is_featured', name: 'is_featured', orderable: false, defaultContent: '' },
            { data: 'is_private', name: 'is_private', orderable: false, defaultContent: '' },
            { data: 'viewed', name: 'viewed', orderable: false, defaultContent: '' },
            { data: 'status', name: 'is_active', searchable: false },
            { data: 'created', name: 'created_at' },
        ],
    });
  </script>
@endpush
```

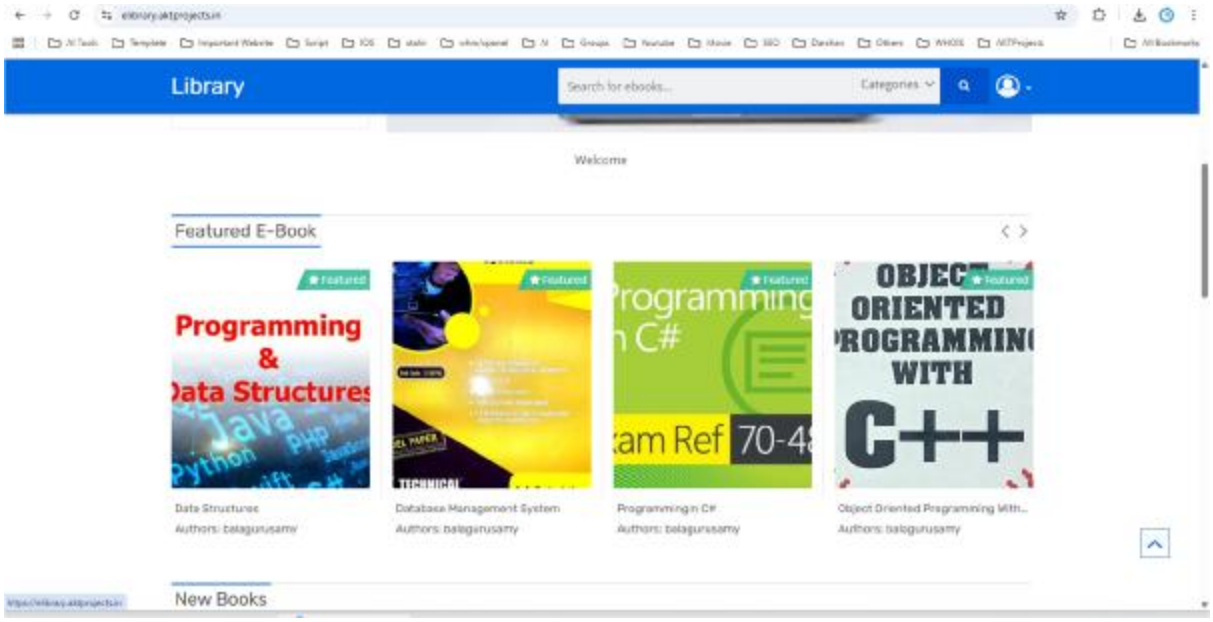
Auth (Login)

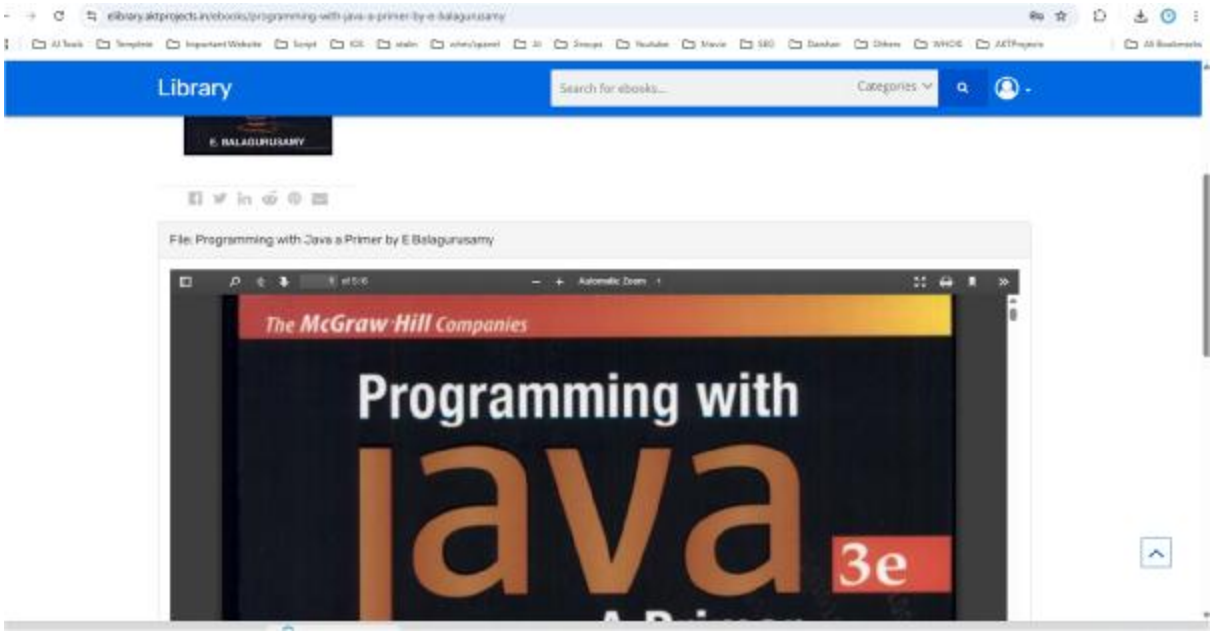
```
@extends('user::admin.auth.layout')

@section('title', clean(trans('user::auth.sign_in')))

@section('content')
  <div class="container container-login">
    @include('admin::include.notification')
    <h3 class="text-center">{{ clean(trans('user::auth.sign_in')) }}</h3>
    <div class="login-form">
      <form method="POST" action="{{ route('admin.login.post') }}">
        {{ csrf_field() }}

        <div class="form-group {{ $errors->has('email') ? 'has-error': '' }}">
          <label for="email">{{ clean(trans('user::auth.email')) }} <span class="required-label">*</span></label>

          <input type="text" name="email" value="{{ old('email') }}" class="form-control" id="email" placeholder="{{ clean(trans('user::attributes.users.email')) }}" autofocus>
          @if($errors->has('email'))
```

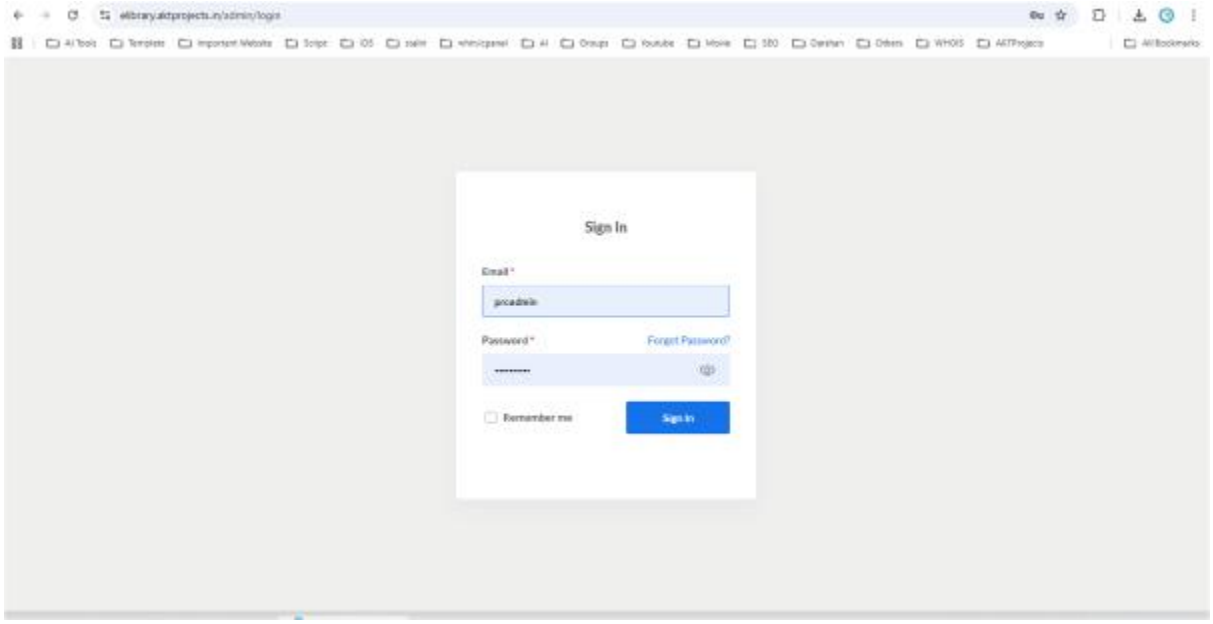
```
                <span class="help-block">{{ clean($errors->first('email')) }}</span>
            @endif

        </div>

        <div class="form-group {{ $errors->has('password') ? 'has-error': '' }}">
                                <label for="password" class="placeholder">{{ clean(trans('user::auth.password')) }} <span class="required-label">*</span></label>
                                <a href="{{ route('admin.reset') }}" class="link float-right">
                {{ clean(trans('user::auth.forgot_password')) }}
            </a>
                                <div class="position-relative">
                                    <input type="password" class="form-control" name="password" placeholder="{{ clean(trans('user::attributes.users.password')) }}">
                                    <div class="show-password">
                                        <i class="icon-eye"></i>
                                    </div>
                                </div>

            @if($errors->has('password'))
                <span class="help-block">{{ clean($errors->first('password')) }}</span>
            @endif
                            </div>

        <div class="form-group form-action-d-flex mb-3">
                                <div class="custom-control custom-checkbox">
```

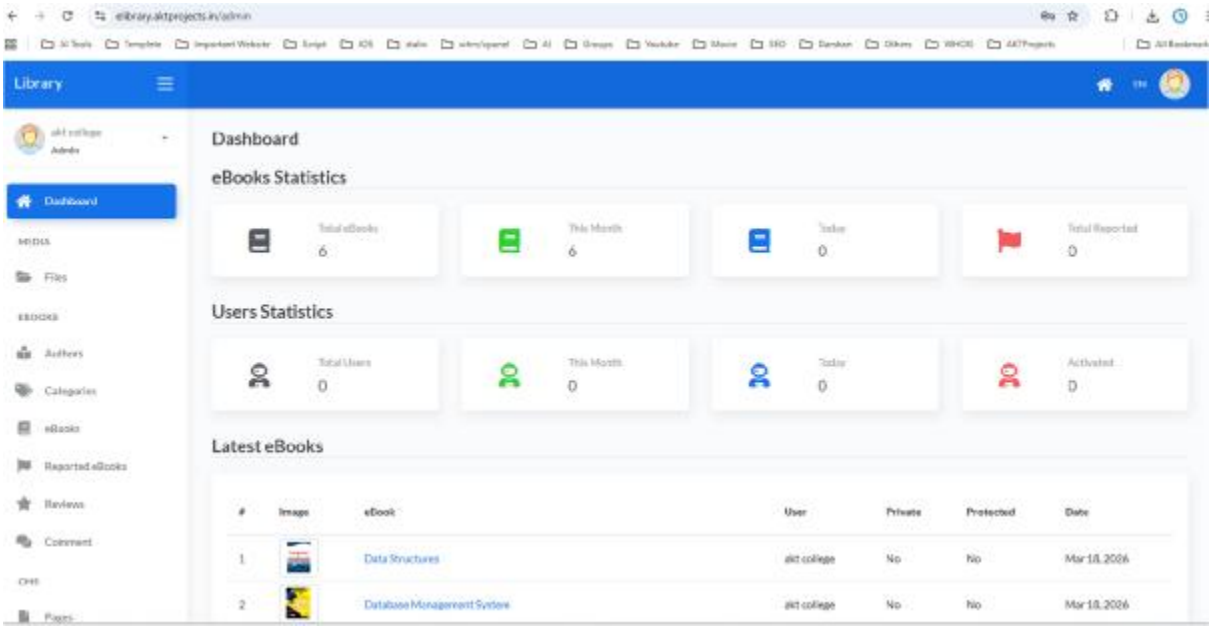
```
                                        <input type="hidden"
name="remember_me" value="0">
                    <input type="checkbox" name="remember_me" value="1"
id="remember_me" class="custom-control-input">
                                        <label class="custom-control-label m-0"
for="remember_me">{{ clean(trans('user::attributes.auth.remember_me')) }}</label>
                                    </div>

                <button type="submit" class="btn btn-primary col-md-5 float-right mt-3
mt-sm-0 fw-bold" data-loading>{{ clean(trans('user::auth.sign_in')) }}</button>

                                </div>
            </form>
        </div>

    </div>
@endsection
```

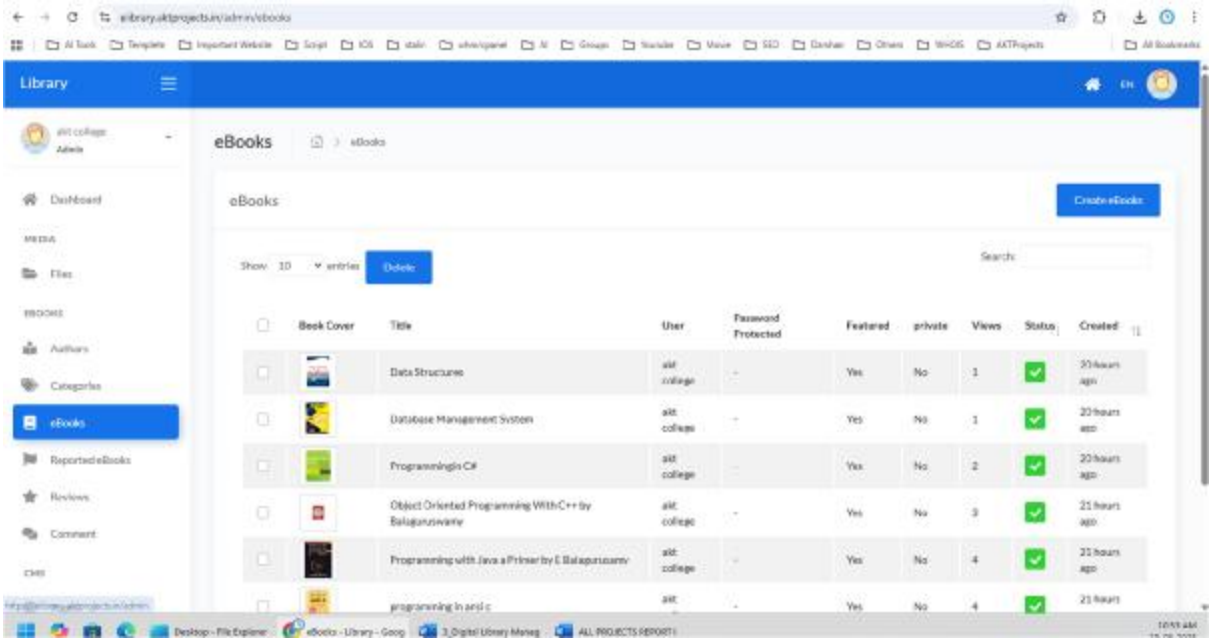
4.4. SCREEN SHOT

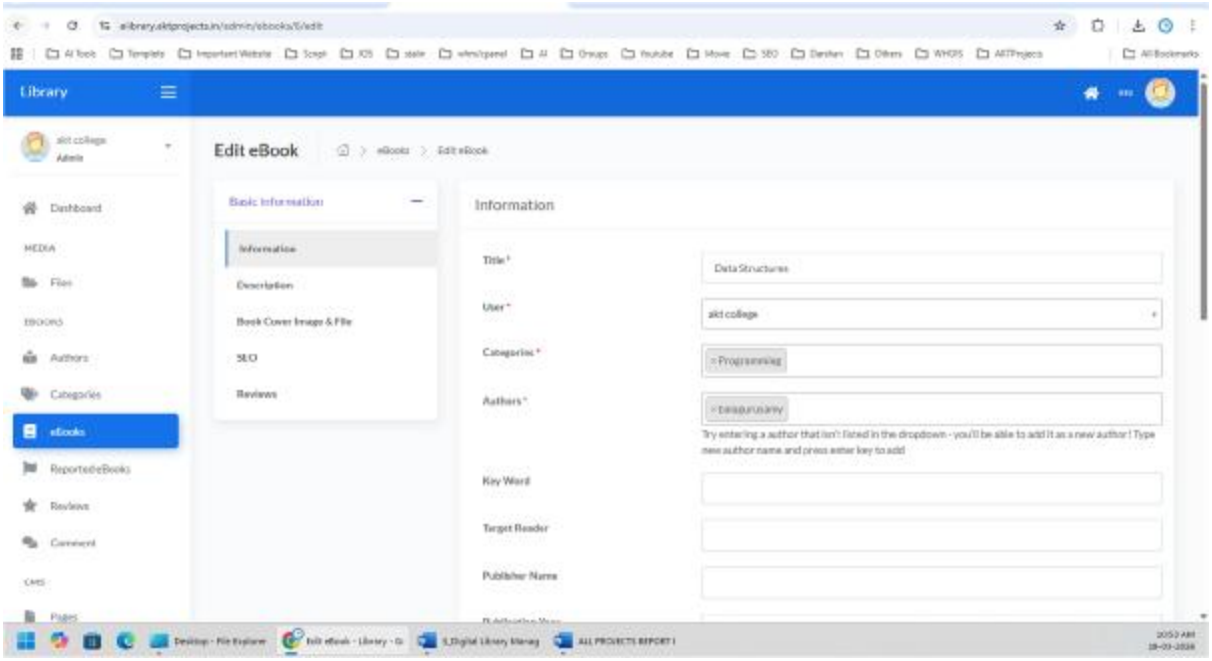


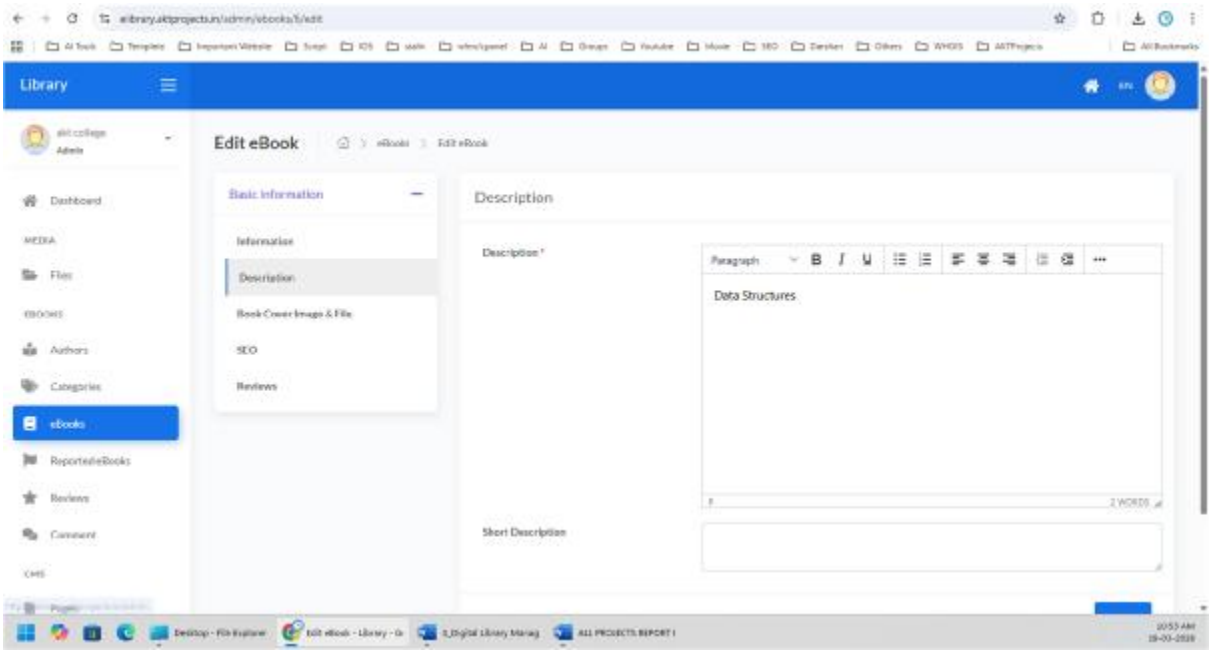

Library
Search for ebooks...
Categories
File: Programming with Java a Primer by E Balagurusamy
The McGraw-Hill Companies
Programming with
Java
3e

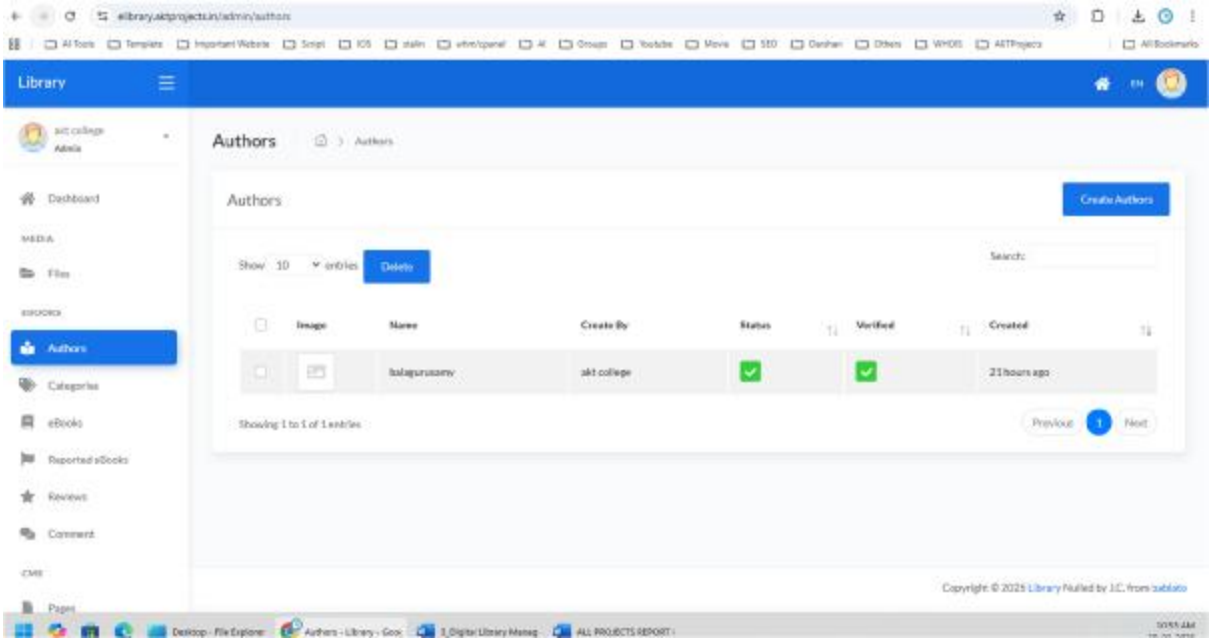

Sign In
Email *
Password *
Forget Password?
Remember me
Sign In

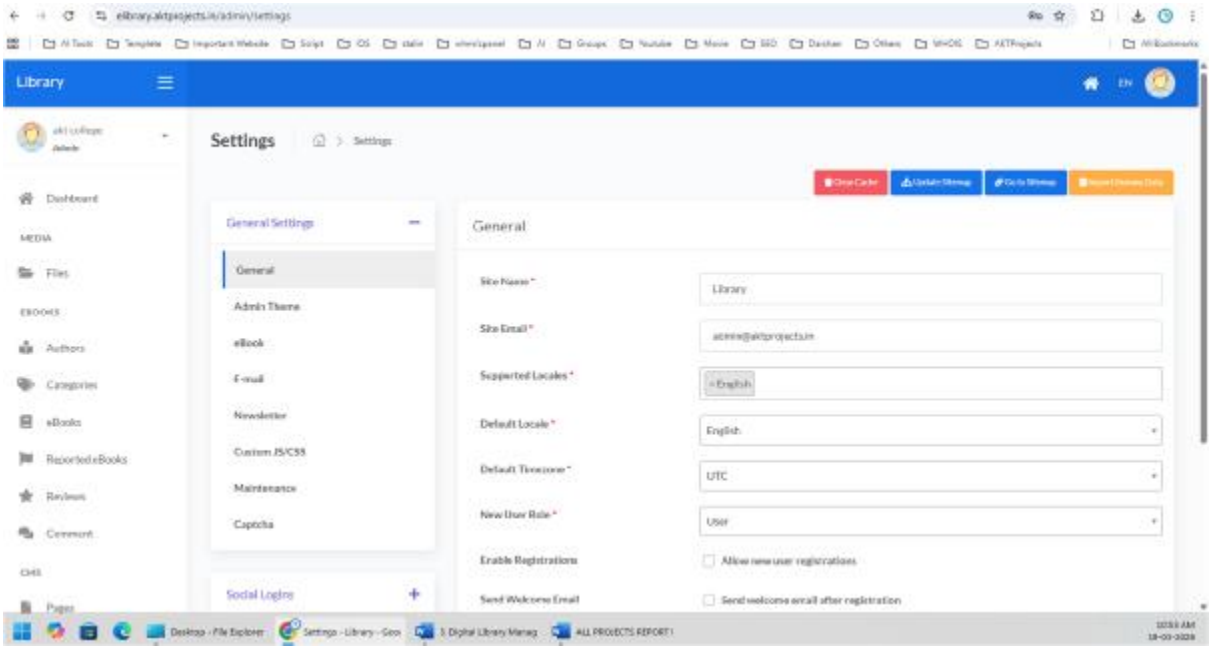

Library
Dashboard
eBooks Statistics
Total eBooks
6
This Month
6
Today
0
Total Reported
0
Users Statistics
Total Users
0
This Month
0
Today
0
Activated
0
Latest eBooks
Image eBook User Private Protected Date
1 Data Structures akt college No No Mar 18, 2026
2 Database Management System akt college No No Mar 18, 2026

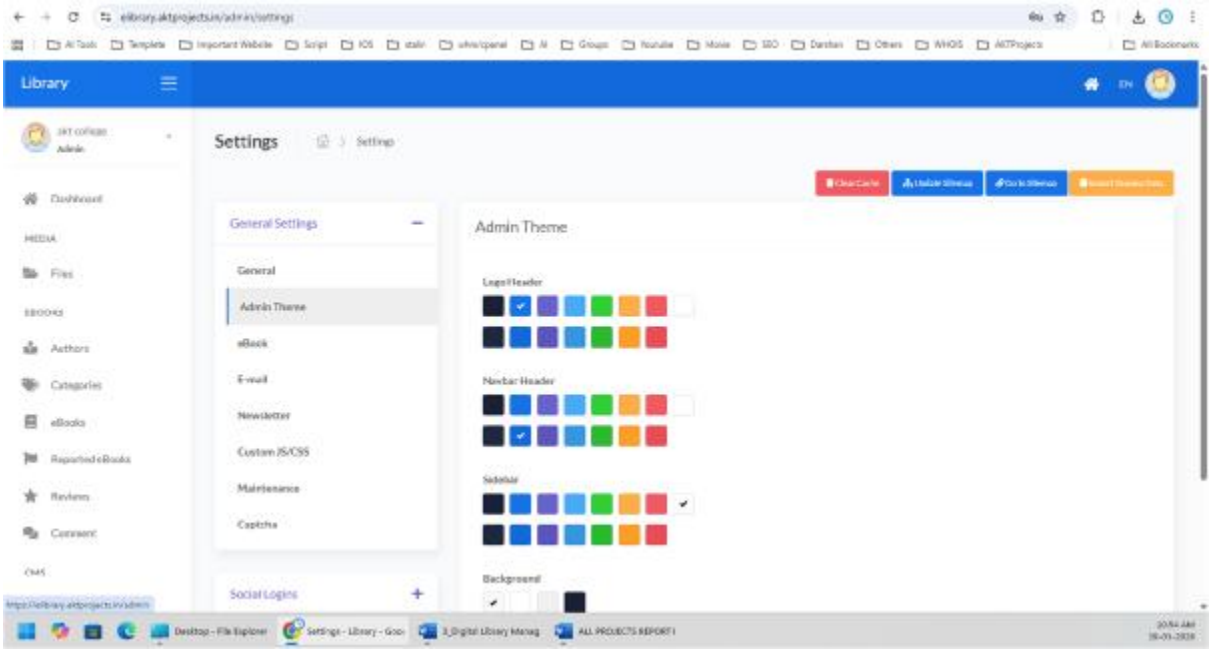

Library
eBooks
Create eBooks
Show 10 entries
Delete
Search:
Book Cover Title User Password Protected Featured private Views Status Created
Data Structures akt college Yes No 1
Database Management System akt college Yes No 1
Programming in C# akt college Yes No 2
Object Oriented Programming With C++ by Balaguruswamy akt college Yes No 3
Programming with Java a Primer by E Balaguruswamy akt college Yes No 4
programming in ansi c akt Yes No 4


Library
Edit eBook
Basic Information
Information
Description
Book Cover Image & File
SEO
Reviews
Title *
Data Structures
User *
akt college
Categories *
Programming
Authors *
Key Word
Target Reader
Publisher Name
Dashboard
Files
Authors
Categories
eBooks
ReportedeBooks
Reviews
Comment
Pages


Library
Edit eBook
Basic Information
Description
Information
Book Cover Image & File
SEO
Reviews
Description *
Paragraph
Data Structures
Short Description
Dashboard
Files
Authors
Categories
eBooks
ReportedeBooks
Reviews
Comment


Library
Authors
Authors
Create Authors
Dashboard
Files
Authors
Categories
eBooks
Reported eBooks
Reviews
Comment
Pages
Image
Name
Create By
Status
Verified
Created
Delete
Search:
Previous
Next


Library
Settings
Dashboard
Files
Authors
Categories
eBooks
Reported eBooks
Reviews
Comment
Pages
General Settings
General
Admin Theme
eBook
E-mail
Newsletter
Custom JS/CSS
Maintenance
Captcha
Social Logins
Site Name *
Site Email *
Supported Locales *
Default Locale *
Default Timezone *
New User Role *
Enable Registrations
Send Welcome Email
Library
English
UTC
User



5. CONCLUSION

The **Digital Library Management with Cloud Storage** system successfully transforms traditional library operations into a modern, efficient, and accessible digital platform. By integrating web technologies with cloud storage, the system enables users to access academic resources anytime and from anywhere, eliminating the limitations of physical libraries.

The implementation ensures **secure user management, efficient resource handling, fast search functionality, and reliable storage of digital content**. The use of cloud storage enhances scalability, data availability, and backup capabilities, while the structured database design ensures organized and quick retrieval of information.

This project demonstrates how technologies like **Laravel, MySQL, and cloud services** can be effectively combined to build a scalable and user-friendly application suitable for educational institutions. Overall, the system improves accessibility, reduces manual effort, and provides a smart solution for managing digital learning resources.

The developed system is reliable, flexible, and capable of handling future expansions, making it a strong foundation for next-generation digital libraries.

REFERENCES

- Ian Sommerville, Software Engineering, 10th Edition, Pearson Education, 2016.
- Roger S. Pressman, Software Engineering: A Practitioner's Approach, McGraw-Hill, 2014.
- E. Balagurusamy, Programming with PHP, McGraw-Hill Education, 2013.
- Luke Welling & Laura Thomson, PHP and MySQL Web Development, Addison-Wesley, 2017.
- Taylor Otwell, Laravel Official Documentation, Laravel Framework, 2024.
- MySQL Documentation, Oracle Corporation, https://dev.mysql.com/doc/
- Amazon Web Services, AWS S3 Documentation, https://aws.amazon.com/s3/
- Google Cloud, Cloud Storage Documentation, https://cloud.google.com/storage

- DigitalOcean, Spaces Object Storage Documentation, https://doc.digitalocean.com/
- MDN Web Docs, HTML, CSS, and JavaScript Documentation, https://developer.mozilla.org/
- W3Schools, Web Development Tutorials, https://www.w3schools.com/
- Bootstrap Documentation, https://getbootstrap.com/docs/
- OWASP Foundation, Web Application Security Guidelines, https://owasp.org/
- Nginx Documentation, https://nginx.org/en/docs/
- Apache HTTP Server Documentation, https://httpd.apache.org/docs/