WEB-BASED STUDENT ACADEMIC PERFORMANCE TRACKING SYSTEM

PROJECT REPORT 2025-2026

Submitted By

24500423 NISHANTH C

24500428 SARANYA M

24590079 NAVIYA T

24590083 VEERAMANI K

Under the guidance of

Mrs S.DEVI B.E.,(CSE)

LECT/IT

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 **Web-Based Student Academic Performance Tracking System** has been submitted **NISHANTH C, SARANYA M, NAVIYA T, VEERAMANI K** 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 S.DEVI 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 S.DEVI 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 Web-Based Student Academic Performance Tracking System is designed to provide an efficient and centralized platform for monitoring, analyzing, and managing students' academic progress. In traditional educational systems, academic records are often maintained manually or through disconnected systems, leading to inefficiencies, data inconsistencies, and difficulty in accessing information. This project aims to overcome these limitations by developing a comprehensive web-based solution that allows institutions to track student performance in real time.

The system enables teachers, administrators, and students to interact within a unified digital environment. Teachers can input marks, attendance, and assessment results, while students can view their academic progress, grades, and feedback instantly. The centralized database ensures that all academic records are stored securely and can be accessed anytime. This improves transparency and reduces dependency on manual record-keeping.

One of the major advantages of this system is its ability to generate analytical reports. These reports help in identifying students' strengths and weaknesses, enabling educators to provide personalized guidance. The system also supports performance comparisons across subjects, semesters, and classes, which assists in decision-making at both individual and institutional levels.

The application is built using modern web technologies such as PHP, MySQL, HTML, CSS, and JavaScript, ensuring high performance and scalability. It can be deployed on a VPS or cloud server, allowing multiple users to access the system simultaneously without performance degradation. The responsive design ensures compatibility across devices, including desktops, tablets, and smartphones.

Security is a key aspect of the system. Role-based access control ensures that only authorized users can access specific functionalities. Data encryption and secure login mechanisms protect sensitive academic information from unauthorized access.

In conclusion, the proposed system provides a reliable, efficient, and scalable solution for academic performance tracking. It enhances communication between stakeholders, improves data accuracy, and supports better educational outcomes through data-driven insights.

1. INTRODUCTION

In the modern education system, tracking student academic performance is essential for ensuring quality education and continuous improvement. Traditional methods of maintaining academic records, such as paper-based systems or isolated software applications, are often inefficient and prone to errors. These systems lack real-time accessibility and make it difficult for stakeholders to monitor progress effectively. The need for a digital solution has led to the development of web-based academic tracking systems.

The Web-Based Student Academic Performance Tracking System is designed to address these challenges by providing a centralized platform for managing academic data. It allows teachers to record marks, attendance, and assessments, while students and administrators can access this information instantly. This real-time accessibility improves transparency and ensures that all stakeholders are informed about academic progress.

The system incorporates role-based access control, ensuring that users can only access functionalities relevant to their roles. For example, teachers can update student records, students can view their performance, and administrators can manage system operations. This enhances both security and usability.

Another key feature of the system is its ability to generate reports and analytics. These reports provide valuable insights into student performance, helping educators identify areas that require improvement. By analyzing trends and patterns, institutions can implement strategies to enhance learning outcomes.

The system is developed using modern web technologies, making it scalable and adaptable to different educational environments. It can be deployed on cloud or VPS servers, ensuring high availability and performance. The responsive design allows users to access the system from various devices, making it convenient and user-friendly.

In summary, the introduction of this system represents a significant advancement in academic management. It simplifies data handling, improves communication, and supports better decision-making, ultimately contributing to enhanced educational quality.

2. LITERATURE SURVEY

. The development of academic performance tracking systems has been influenced by various research studies and technological advancements. Early systems were primarily manual, relying on physical records and spreadsheets. These methods were time-consuming, prone to errors, and lacked scalability. Researchers have emphasized the need for automated systems to improve efficiency and accuracy in academic management.

Several studies have explored the use of database management systems in education. Relational databases such as MySQL have been widely adopted for storing academic records due to their reliability and structured data handling capabilities. These systems ensure data consistency and allow efficient retrieval of information, which is essential for large educational institutions.

Web-based applications have gained popularity due to their accessibility and flexibility. Research indicates that web-based systems provide better user experience compared to standalone applications, as they can be accessed from anywhere using a browser. This has led to the widespread adoption of web technologies such as PHP, HTML, CSS, and JavaScript in educational systems.

Security is another critical aspect highlighted in the literature. With the increasing use of digital systems, protecting sensitive student data has become a priority. Techniques such as encryption, authentication, and role-based access control have been extensively studied and implemented to ensure data security.

Recent advancements in data analytics have also influenced the development of academic tracking systems. Researchers have focused on using analytical tools to evaluate student performance and identify patterns. These insights help educators provide targeted support and improve overall learning outcomes.

Cloud computing has further enhanced the capabilities of academic systems by providing scalability and cost-effectiveness. Cloud-based solutions allow institutions to handle large amounts of data without investing heavily in infrastructure.

In conclusion, the literature survey highlights the importance of web technologies, database systems, security measures, and data analytics in developing effective academic performance tracking systems. These concepts have been incorporated into the proposed system to ensure it meets modern educational

3. PROPOSED SYSTEM

The proposed Web-Based Student Academic Performance Tracking System is designed to provide a comprehensive solution for managing and analyzing student academic data. It aims to eliminate the limitations of traditional systems by offering a centralized, automated, and user-friendly platform.

The system allows teachers to input and manage student marks, attendance, and assessment results. This data is stored in a centralized database, ensuring consistency and easy access. Students can log in to view their academic performance, including grades, attendance records, and feedback from teachers. Administrators can oversee the entire system, manage users, and generate reports.

One of the key features of the system is its reporting and analytics capability. The system generates detailed reports that help educators understand student performance trends. These reports can be used to identify students who need additional support and to evaluate the effectiveness of teaching methods.

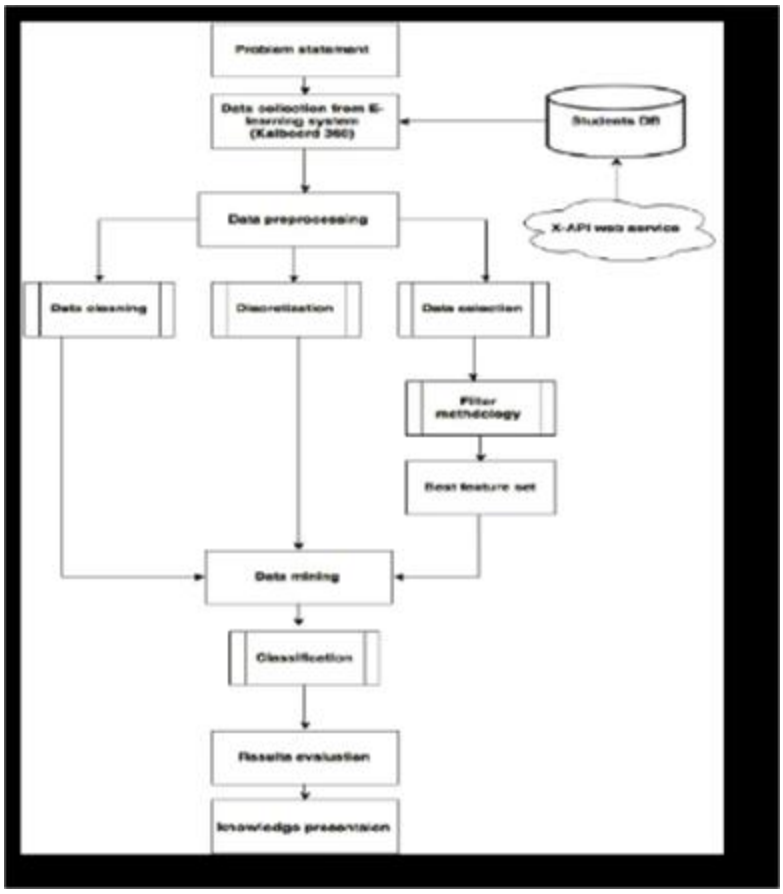
The system also includes a notification feature that alerts students and teachers about important updates, such as exam schedules, results, and deadlines. This improves communication and ensures that users are always informed.

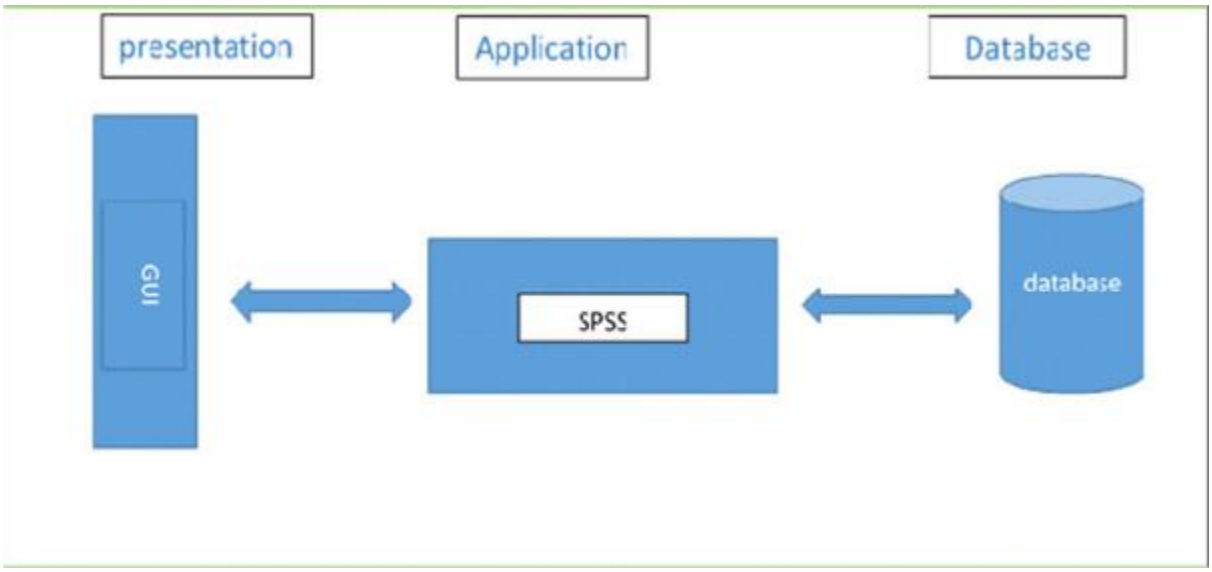
Scalability is a major advantage of the proposed system. It is designed using a modular architecture, allowing new features to be added بسهولة without affecting existing functionalities. The system can be deployed on cloud or VPS servers, ensuring high performance and availability.

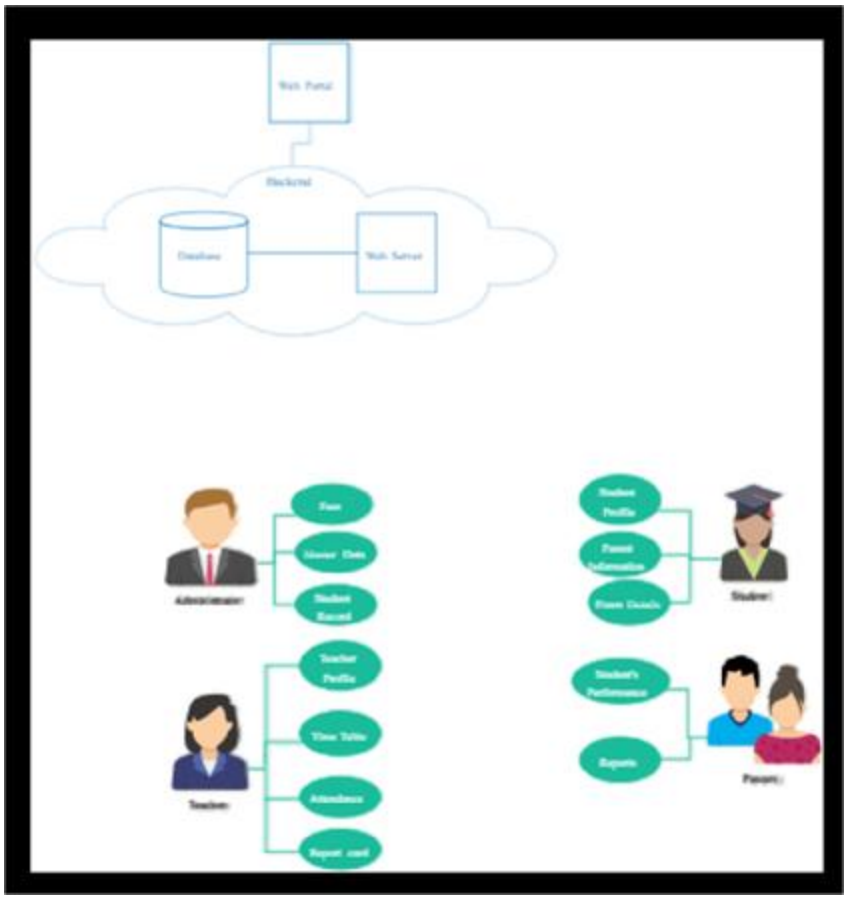
Security measures such as authentication, authorization, and data encryption are implemented to protect sensitive information. Role-based access control ensures that users can only access functionalities relevant to their roles.

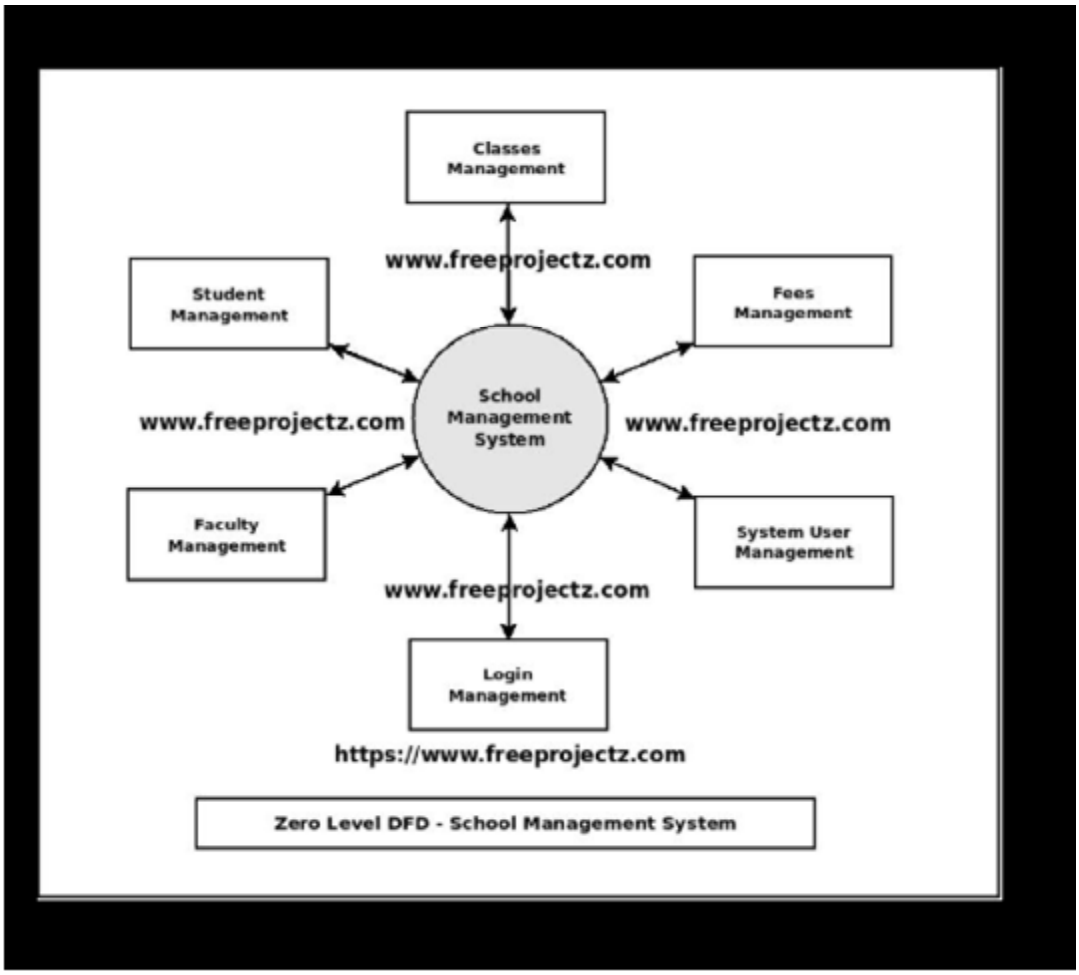
In summary, the proposed system provides an efficient, secure, and scalable solution for academic performance tracking. It enhances data management, improves communication, and supports better decision-making in educational institutions.

3.1 ARCHITECTURE









The architecture of the Web-Based Student Academic Performance Tracking System follows a three-tier architecture, ensuring separation of concerns and efficient system performance. The architecture consists of the presentation layer, application layer, and database layer.

The presentation layer is responsible for the user interface. It is developed using HTML, CSS, and JavaScript, providing a responsive and interactive experience. Users such as students, teachers, and administrators interact with the system through this layer using web browsers.

The application layer handles the business logic of the system. It processes user requests, performs validations, and communicates with the database. This layer is implemented using PHP or Laravel, ensuring efficient handling of system functionalities such as data processing, report generation, and authentication.

The database layer stores all academic data, including student details, marks, attendance, and reports. MySQL is used as the database management system, providing reliable and structured data storage. This layer ensures data integrity and supports efficient data retrieval.

The system also includes security components such as authentication and authorization mechanisms. These ensure that only authorized users can access specific features. Data encryption and secure communication protocols further enhance system security.

Additionally, the system can be deployed on a VPS or cloud server, ensuring scalability and high availability. Load balancing and backup mechanisms can be implemented to handle large user traffic and prevent data loss.

The modular design of the architecture allows easy maintenance and future enhancements. Each component operates independently, making it easier to update or replace without affecting the entire system.

3.2 MODULES DESCRIPTION

1. User Management Module

The User Management Module is responsible for handling all user-related activities within the system, including registration, login, authentication, and authorization. It ensures that users such as students, teachers, and administrators can securely access the system based on their roles. During registration, users provide essential details like name, email, and password, which are securely stored using encryption techniques.

This module implements role-based access control, ensuring that each user can only access features relevant to their role. For example, students can view their academic records, teachers can update marks and attendance, and administrators have full control over the system. This separation of roles enhances both security and usability.

Additionally, the module supports account management features such as password reset, profile updates, and account activation or deactivation. Administrators can monitor user activity and manage accounts efficiently. This module forms the backbone of system security, ensuring that sensitive academic data is protected from unauthorized access.

2. Student Profile Module

The Student Profile Module maintains comprehensive records of each student in the system. It stores personal details such as name, registration number, contact information, course, and department. This module acts as a centralized database for all student-related information, ensuring easy access and data consistency.

Administrators and authorized staff can create, update, and manage student profiles efficiently. The module integrates with other components such as marks and attendance modules to provide a complete overview of a student's academic performance. This integration eliminates redundancy and ensures accurate data representation.

Students can also view their profiles, allowing them to verify their information and stay updated. The module supports data validation to ensure that only accurate and relevant information is stored. Overall, it plays a crucial role in organizing and managing student data effectively.

3. Marks Management Module

The Marks Management Module allows teachers to record, update, and manage student marks for assignments, tests, and examinations. It ensures that academic performance data is stored accurately and can be accessed whenever needed. Teachers can enter marks for different subjects and assessments through an easy-to-use interface. This module includes validation mechanisms to prevent errors during data entry. It also supports editing and updating marks, ensuring flexibility in case of corrections. The stored data is used by other modules, such as performance analysis and report generation, to provide meaningful insights.

Students can view their marks in real time, which enhances transparency and helps them track their progress. The module also allows administrators to monitor grading activities. Overall, it plays a key role in maintaining accurate academic records and supporting performance evaluation.

4. Attendance Tracking Module

The Attendance Tracking Module is designed to record and manage student attendance efficiently. Teachers can mark attendance daily or for specific sessions, and the system automatically calculates attendance percentages. This helps in monitoring student participation and identifying attendance patterns.

The module highlights students with low attendance, enabling teachers and administrators to take corrective actions. It also provides attendance reports that can be used for academic evaluations and compliance purposes. Automated calculations reduce manual effort and minimize errors.

Students can view their attendance records, allowing them to stay informed and improve their attendance if needed. The module integrates with the performance system, as attendance often impacts academic performance. Overall, it enhances discipline and ensures accurate attendance management.

5. Performance Analysis Module

The Performance Analysis Module processes academic data to evaluate student performance. It analyzes marks, attendance, and other metrics to identify trends,

strengths, and weaknesses. This helps educators understand how students are performing across different subjects and time periods.

The module generates visual representations such as charts and summaries, making it easier to interpret data. Teachers can use these insights to provide targeted support to students who may be struggling. It also helps in identifying high-performing students and encouraging them further.

Administrators can use performance analysis to evaluate overall academic standards and improve teaching strategies. This module plays a crucial role in data-driven decision-making, ensuring that educational outcomes are continuously improved.

6. Report Generation Module

The Report Generation Module creates detailed academic reports for students. These reports include mark sheets, progress reports, and performance summaries. The module ensures that all relevant data is compiled and presented in a structured format. Reports can be generated for individual students, classes, or entire departments. This helps teachers and administrators analyze performance at different levels. The module supports exporting reports in formats such as PDF for easy sharing and documentation. Students and parents can access reports to understand academic progress. The module enhances transparency and communication between stakeholders. Overall, it simplifies the process of generating and sharing academic reports.

7. Notification Module

The Notification Module ensures effective communication within the system by sending alerts and updates to users. Notifications can include exam schedules, result announcements, attendance warnings, and important messages from administrators.
The module supports multiple delivery methods such as system alerts, emails, or SMS notifications. This ensures that users receive timely information and do not miss important updates. Notifications can also be customized based on user roles and preferences.

By keeping users informed, the module improves engagement and reduces communication gaps. It plays a vital role in maintaining smooth system operations and ensuring that all stakeholders are aware of important events.

8. Admin Dashboard Module

The Admin Dashboard Module provides administrators with a centralized interface to manage the entire system. It offers tools for user management, data monitoring, and report generation. Administrators can view system statistics, track user activities, and ensure smooth operation.

The dashboard presents data in an organized and visually appealing manner, making it easy to understand system performance. Administrators can also configure system settings and manage access permissions.

This module ensures that the system runs efficiently and that any issues are addressed promptly. It acts as the control center of the application, providing complete oversight and management capabilities.

9. Search and Filter Module

The Search and Filter Module enhances system usability by allowing users to quickly locate specific data. Users can search for student records, marks, or attendance using keywords or filters such as name, ID, or course.

Advanced filtering options enable users to narrow down results based on multiple criteria. This saves time and improves efficiency, especially in large databases. The module ensures fast and accurate retrieval of information.

By simplifying data access, this module improves user experience and productivity. It is an essential component for handling large volumes of academic data.

10. Security Module

The Security Module ensures that all data within the system is protected from unauthorized access and threats. It implements authentication, authorization, and encryption techniques to safeguard sensitive information.

The module includes secure login mechanisms, session management, and data encryption to prevent breaches. Role-based access control ensures that users can only access permitted functionalities.

Regular monitoring and backup features are also included to prevent data loss and ensure system reliability. This module is critical for maintaining trust and ensuring the integrity of academic data.

4. IMPLEMENTATION

The implementation of the Web-Based Student Academic Performance Tracking System involves the practical development of the proposed modules using web technologies, database integration, and secure deployment methods. The system is developed as a browser-based application so that students, teachers, and administrators can access it easily from any device connected to the internet or local network. The implementation process begins with requirement analysis, where the major academic activities such as student registration, marks entry, attendance management, report generation, and performance monitoring are identified and converted into system functions. Based on these requirements, the database structure, user roles, interface flow, and backend logic are designed carefully to ensure smooth operation.

The frontend of the system is implemented using HTML, CSS, Bootstrap, and JavaScript to provide a responsive and user-friendly interface. Each page is designed with simple navigation so that users can access their required features without confusion. Students are provided with pages to view marks, attendance, subject-wise progress, and report cards. Teachers are given dashboards to enter marks, update attendance, and monitor class performance. Administrators are provided with a complete control panel to manage users, academic records, and system settings. JavaScript is used to improve interactivity, validate form fields, and reduce input errors before data is submitted to the server.

The backend of the system is implemented using PHP or a framework such as Laravel, which handles the business logic and communication between the user interface and the database. Whenever a user performs an action, such as logging in, entering marks, or generating a report, the backend processes the request, validates the input, and performs the necessary database operations. Role-based authentication is implemented so that users are redirected to their respective dashboards after login. Teachers can only access academic entry and update functions, students can only view their own data, and administrators can manage the full system. This separation ensures both security and operational clarity.

The database is implemented using MySQL, where tables are created for students, teachers, subjects, classes, marks, attendance, semesters, login details, and reports. Relationships are established using primary keys and foreign keys so that data remains connected and consistent. For example, each mark entry is linked to a specific student, subject, and semester. Attendance records are mapped to students and dates, allowing the system to calculate percentages automatically. The use of a structured relational database ensures efficient storage, retrieval, and updating of academic records without duplication or inconsistency.

Security implementation is an important part of the system. User passwords are stored in encrypted format, and session management is used to protect authenticated access. Input validation and sanitization techniques are applied to prevent SQL injection, unauthorized form submission, and invalid data entry. Access control is enforced at every level so that no user can access restricted pages directly. Backup mechanisms can also be implemented to preserve academic data in case of accidental deletion or server failure. These security practices make the application reliable for institutional use.

Finally, the completed system is deployed on a local server, VPS server, or cloud hosting environment depending on institutional needs. After deployment, testing is conducted to verify all modules, including login, marks entry, attendance, analytics, report generation, and notifications. Errors identified during testing are corrected to improve performance and usability. With proper implementation, the system provides an efficient digital platform for managing academic performance, reducing manual work, improving transparency, and supporting better educational decision-making.

4.1 SOFTWARE REQUITEMENTS

1. Operating System

- Windows 10 / 11 (for development)
- Linux (Ubuntu/CentOS) recommended for production server

2. Web Server

- Apache Server (with mod_rewrite enabled)

  OR

- Nginx Server

3. Backend Technology

- PHP >= 8.2
- Laravel Framework (Latest Version)

4. Database

- MySQL Server (5.7 or higher)

  OR

- MariaDB

5. Frontend Technology

- HTML5
- CSS3
- JavaScript
- Bootstrap (for responsive UI design)
- Blade Template Engine (Laravel)

6. Required PHP Extensions

- PDO PHP Extension
- OpenSSL PHP Extension
- Mbstring PHP Extension
- Exif PHP Extension
- Fileinfo Extension
- XML PHP Extension
- Ctype PHP Extension
- JSON PHP Extension
- Tokenizer PHP Extension

- cURL PHP Extension

7. Additional Tools & Software

- Composer (Dependency Manager for PHP)
- Git (Version Control System)
- Node.js & NPM (for frontend asset compilation, optional)

8. Server Configuration

- Enable **mod_rewrite** (Apache)
- Enable **HTTPS (SSL Certificate)** for secure communication
- Configure .env file for database and app settings
- Set proper file permissions for storage and cache

9. Browser Compatibility

- Google Chrome
- Mozilla Firefox
- Microsoft Edge
- Safari

10. Hosting Environment

- VPS Server / Cloud Hosting (AWS, DigitalOcean, etc.)
- Minimum recommended configuration:
    - 2+ CPU Cores
    - 4 GB RAM
    - 50 GB Storage

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
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

```
@extends('student.layouts.master')
@section('title', $title)
@section('content')

<!-- Start Content-->
<div class="main-body">
   <div class="page-wrapper">
      <!-- [ Main Content ] start -->
      <div class="row">
         <div class="col-sm-12">
            <div class="card">
               <div class="card-header">
                  <h5>{{ $title }}</h5>
               </div>
               <div class="card-block">
                  <form class="needs-validation" novalidate method="get" action="{{ route($route.'.index') }}">
                     <div class="row gx-2">
                        <div class="form-group col-md-3">
                           <label for="type">{{ __('field_exam') }} {{ __('field_type') }} <span>*</span></label>
                           <select class="form-control" name="type" id="type" required>
                              <option value="">{{ __('select') }}</option>
                              @foreach( $types as $type )
                              <option value="{{ $type->id }}" @if( $selected_type == $type->id) selected @endif>{{ $type->title }}</option>
                              @endforeach
                           </select>
```

```
                    <div class="invalid-feedback">
                    {{ __('required_field') }} {{ __('field_exam') }} {{ __('field_type') }}
                    </div>
                  </div>
                  <div class="form-group col-md-3">
                    <button type="submit" class="btn btn-info btn-filter"><i class="fas fa-search"></i> {{ __('btn_search') }}</button>
                  </div>
                </div>
              </form>
            </div>

            @if(isset($rows))
            <div class="card-block">
              <!-- [ Data table ] start -->
              <div class="table-responsive">
                <table id="export-table" class="display table nowrap table-striped table-hover" style="width:100%">
                  <thead>
                    <tr>
                      <th>#</th>
                      <th>{{ __('field_subject') }}</th>
                      <th>{{ __('field_teacher') }}</th>
                      <th>{{ __('field_room') }}</th>
                      <th>{{ __('field_date') }}</th>
                      <th>{{ __('field_start_time') }}</th>
                      <th>{{ __('field_end_time') }}</th>
                    </tr>
                  </thead>
                  <tbody>
```

```
@foreach($rows as $key => $row)
<tr>
  <td>{{ $key + 1 }}</td>
  <td>{{ $row->subject->code ?? '' }} - {{ $row->subject->title ?? '' }}</td>
  <td>
    @foreach($row->users as $teacher)
    {{ $teacher->first_name }} {{ $teacher->last_name }}@if($loop->last) @else , @endif<br/>
    @endforeach
  </td>
  <td>
    @foreach($row->rooms as $room)
    {{ $room->title }}@if($loop->last) @else , @endif
    @endforeach
  </td>
  <td>
    @if(isset($setting->date_format))
    {{ date($setting->date_format, strtotime($row->date)) }}
    @else
    {{ date("Y-m-d", strtotime($row->date)) }}
    @endif
  </td>
  <td>
    @if(isset($setting->time_format))
    {{ date($setting->time_format, strtotime($row->start_time)) }}
    @else
    {{ date("h:i A", strtotime($row->start_time)) }}
    @endif
  </td>
```

```
                        <td>
                            @if(isset($setting->time_format))
                            {{ date($setting->time_format, strtotime($row->end_time)) }}
                            @else
                            {{ date("h:i A", strtotime($row->end_time)) }}
                            @endif
                        </td>
                    </tr>
                    @endforeach
                </tbody>
            </table>
        </div>
        <!-- [ Data table ] end -->
    </div>
    @endif

    </div>
  </div>
 </div>
 <!-- [ Main Content ] end -->
</div>
</div>
<!-- End Content-->

@endsection
```

Assessment

```
@extends('student.layouts.master')
@section('title', $title)
@section('content')

<!-- Start Content-->
<div class="main-body">
   <div class="page-wrapper">
      <!-- [ Main Content ] start -->
      <div class="row">
         <div class="col-sm-12">
            <div class="card">
               <div class="card-header">
                  <h5>{{ $title }} {{ __('list') }}</h5>
               </div>
               <div class="card-block">
                  <form class="needs-validation" novalidate method="get" action="{{ route($route .'.index') }}">
                     <div class="row gx-2">
                        <div class="form-group col-md-3">
                           <label for="session">{{ __('field_session') }}</label>
                           <select class="form-control" name="session" id="session">
                              <option value="0">{{ __('all') }}</option>
                              @foreach( $sessions as $session )
                              <option value="{{ $session->session_id }}" @if( $selected_session == $session->session_id) selected @endif>{{ $session->session->title }}</option>
                              @endforeach
                           </select>

                           <div class="invalid-feedback">
```

```
                        {{ __('required_field') }} {{ __('field_session') }}
                    </div>
                </div>
                <div class="form-group col-md-3">
                    <label for="semester">{{ __('field_semester') }}</label>
                    <select class="form-control" name="semester" id="semester">
                        <option value="0">{{ __('all') }}</option>
                        @foreach( $semesters as $semester )
                        <option value="{{ $semester->semester_id }}" @if( $selected_semester == $semester->semester_id) selected @endif>{{ $semester->semester->title }}</option>
                        @endforeach
                    </select>

                    <div class="invalid-feedback">
                        {{ __('required_field') }} {{ __('field_semester') }}
                    </div>
                </div>

                <div class="form-group col-md-3">
                    <button type="submit" class="btn btn-info btn-filter"><i class="fas fa-search"></i> {{ __('btn_filter') }}</button>
                </div>
            </div>
        </form>
    </div>
    <div class="card-block">
        <!-- [ Data table ] start -->
        <div class="table-responsive">
            <table id="basic-table" class="display table nowrap table-striped table-hover" style="width:100%">
```

```
                    <thead>
                      <tr>
                        <th>#</th>
                        <th>{{ __('field_title') }}</th>
                        <th>{{ __('field_subject') }}</th>
                        <th>{{ __('field_session') }}</th>
                        <th>{{ __('field_semester') }}</th>
                        <th>{{ __('field_start_date') }}</th>
                        <th>{{ __('field_end_date') }}</th>
                        <th>{{ __('field_status') }}</th>
                      </tr>
                    </thead>
                    <tbody>
                     @foreach( $rows as $key => $row )
                     @if($row->assignment->status == 1)
                      <tr>
                        <td>{{ $key + 1 }}</td>
                        @php
                        $unread = 0;
                        $user = Auth::guard('student')->user();
                        foreach ($user->unreadNotifications as $notification) {
                          if($notification->data['type'] == 'assignment' &&
$notification->data['id'] == $row->assignment->id) {
                            $unread = 1;
                          }
                        }
                        @endphp
                        <td>
                          @if($unread == 1)
                          <a href="{{ route($route.'.show', $row->id) }}"><b>{!!
str_limit($row->assignment->title ?? '', 50, ' ...') !!}</b></a>
```

```
                        @else
                        <a href="{{ route($route.'.show', $row->id) }}">{!!
str_limit($row->assignment->title ?? '', 50, ' ...') !!}</a>
                        @endif
                    </td>
                    <td>{{ $row->assignment->subject->code ?? '' }}</td>
                    <td>{{ $row->studentEnroll->session->title ?? '' }}</td>
                    <td>{{ $row->studentEnroll->semester->title ?? '' }}</td>
                    <td>
                        @if(isset($setting->date_format))
                        {{ date($setting->date_format, strtotime($row-
>assignment->start_date)) }}
                        @else
                        {{ date("Y-m-d", strtotime($row->assignment-
>start_date)) }}
                        @endif
                    </td>
                    <td>
                        @if(isset($setting->date_format))
                        {{ date($setting->date_format, strtotime($row-
>assignment->end_date)) }}
                        @else
                        {{ date("Y-m-d", strtotime($row->assignment->end_date))
}}
                        @endif
                    </td>
                    <td>
                        @if( $row->attendance == 1 )
                        <span class="badge badge-pill badge-success">{{
__('status_submitted') }}</span>
                        @else
```

```
                                <span class="badge badge-pill badge-primary">{{
__('status_pending') }}</span>
                                @endif
                            </td>
                        </tr>
                        @endif
                        @endforeach
                    </tbody>
                </table>
            </div>
            <!-- [ Data table ] end -->
        </div>
    </div>
</div>
<!-- [ Main Content ] end -->
</div>
</div>
<!-- End Content-->

@endsection
```

Show

```
@extends('student.layouts.master')
@section('title', $title)
@section('content')

<!-- Start Content-->
<div class="main-body">
   <div class="page-wrapper">
      <!-- [ Main Content ] start -->
      <div class="row">
         <div class="col-sm-8">
            <div class="card">
               <div class="card-header">
                  <h5>{{ $row->assignment->title ?? '' }}</h5>
               </div>
               <div class="card-block">
                  <a href="{{ route($route.'.index') }}" class="btn btn-primary"><i class="fas fa-arrow-left"></i> {{ __('btn_back') }}</a>
               </div>
               <div class="card-block">
               <!-- Details View Start -->
               <div class="">
                  <div class="row">
                     <div class="col-md-6">
                        <p><mark class="text-primary">{{ __('field_subject') }}:</mark> {{ $row->assignment->subject->code ?? '' }} - {{ $row->assignment->subject->title ?? '' }}</p><hr/>
                        <p><mark class="text-primary">{{ __('field_session') }}:</mark> {{ $row->studentEnroll->session->title ?? '' }}</p><hr/>
                        <p><mark class="text-primary">{{ __('field_semester') }}:</mark> {{ $row->studentEnroll->semester->title ?? '' }}</p><hr/>
```

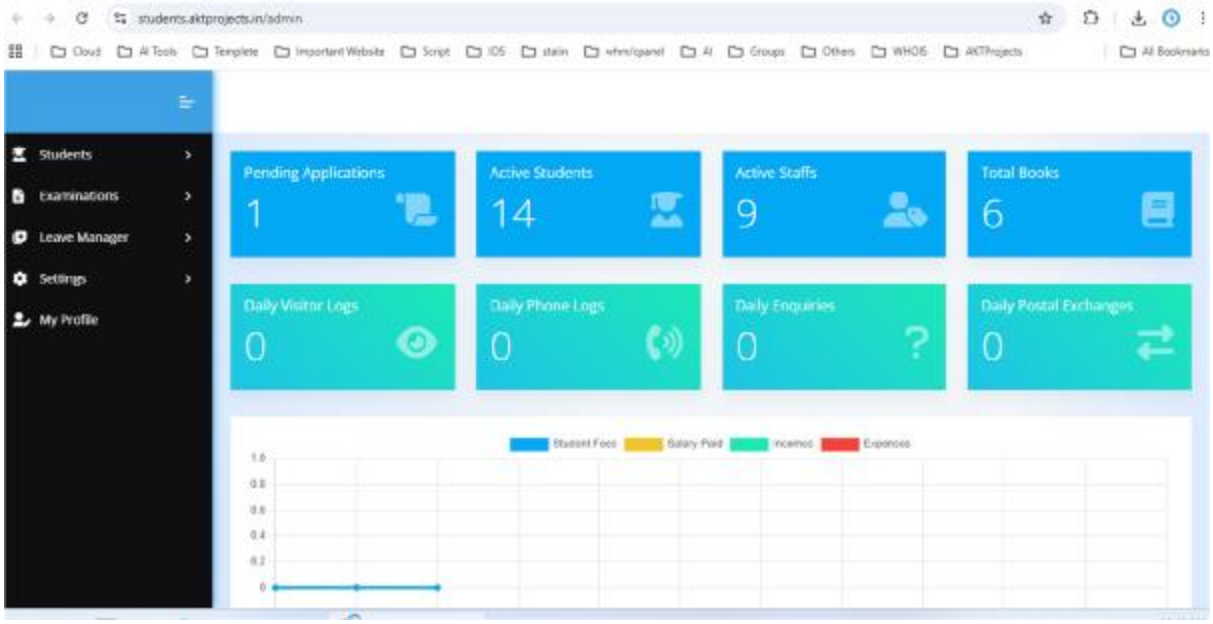
```
                    <p><mark class="text-primary">{{ __('field_section') }}:</mark>
{{ $row->studentEnroll->section->title ?? '' }}</p><hr/>
                </div>
                <div class="col-md-6">
                    @if(!empty($row->marks))
                    <p><mark class="text-primary">{{ __('field_marks_obtained')
}}:</mark> {{ round($row->marks, 2) }}</p><hr/>
                    @endif

                    <p><mark class="text-primary">{{ __('field_max_marks')
}}:</mark> {{ round($row->assignment->total_marks ?? 0, 2) }}</p><hr/>

                    <p><mark class="text-primary">{{ __('field_start_date')
}}:</mark>
                        @if(isset($setting->date_format))
                        {{ date($setting->date_format, strtotime($row->assignment-
>start_date)) }}
                        @else
                        {{ date("Y-m-d", strtotime($row->assignment->start_date)) }}
                        @endif
                    </p><hr/>
                    <p><mark class="text-primary">{{ __('field_end_date')
}}:</mark>
                        @if(isset($setting->date_format))
                        {{ date($setting->date_format, strtotime($row->assignment-
>end_date)) }}
                        @else
                        {{ date("Y-m-d", strtotime($row->assignment->end_date)) }}
                        @endif
                    </p><hr/>
```

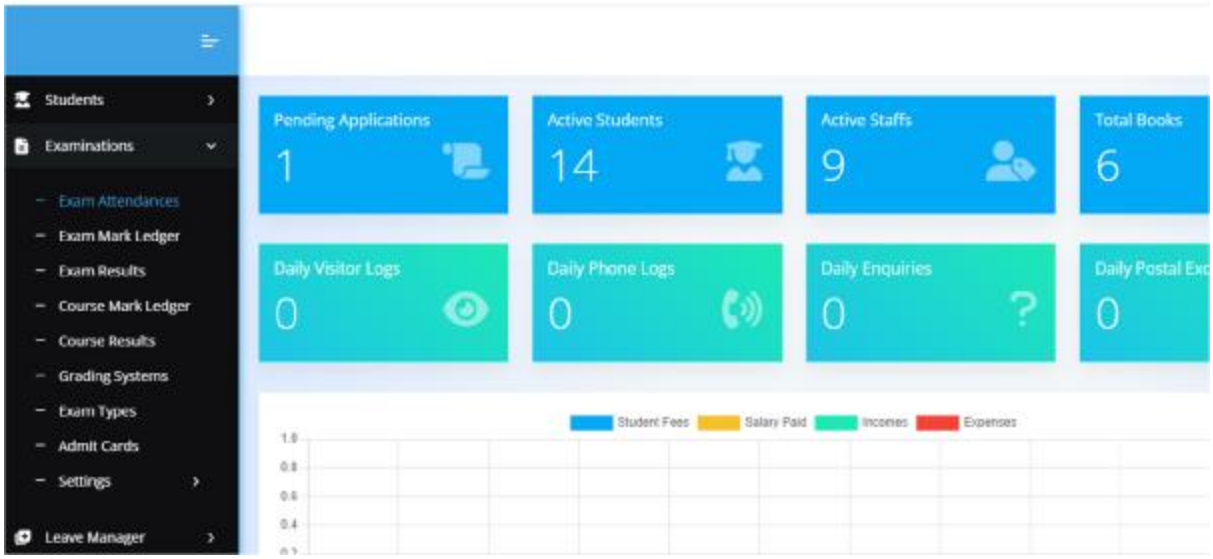
```
                    @if(!empty($row->date))
                    <p><mark class="text-primary">{{ __('field_submission') }} {{
__('field_date') }}:</mark>
                        @if(isset($setting->date_format))
                        {{ date($setting->date_format, strtotime($row->date)) }}
                        @else
                        {{ date("Y-m-d", strtotime($row->date)) }}
                        @endif
                    </p><hr/>
                    @endif

                    <p><mark class="text-primary">{{ __('field_status') }}:</mark>
                        @if( $row->attendance == 1 )
                        <span class="badge badge-pill badge-success">{{
__('status_submitted') }}</span>
                        @else
                        <span class="badge badge-pill badge-primary">{{
__('status_pending') }}</span>
                        @endif
                    </p><hr/>

                    @if(is_file('uploads/'.$path.'/'.$row->assignment->attach))
                    <mark class="text-primary">{{ __('field_attach') }}:</mark>
                    <a href="{{ asset('uploads/'.$path.'/'.$row->assignment->attach)
}}" class="btn btn-dark" download><i class="fas fa-download"></i> {{
__('field_attach') }}</a>
                    @endif
                </div>
            </div>
            <div class="row">
                <div class="col-md-12">
```

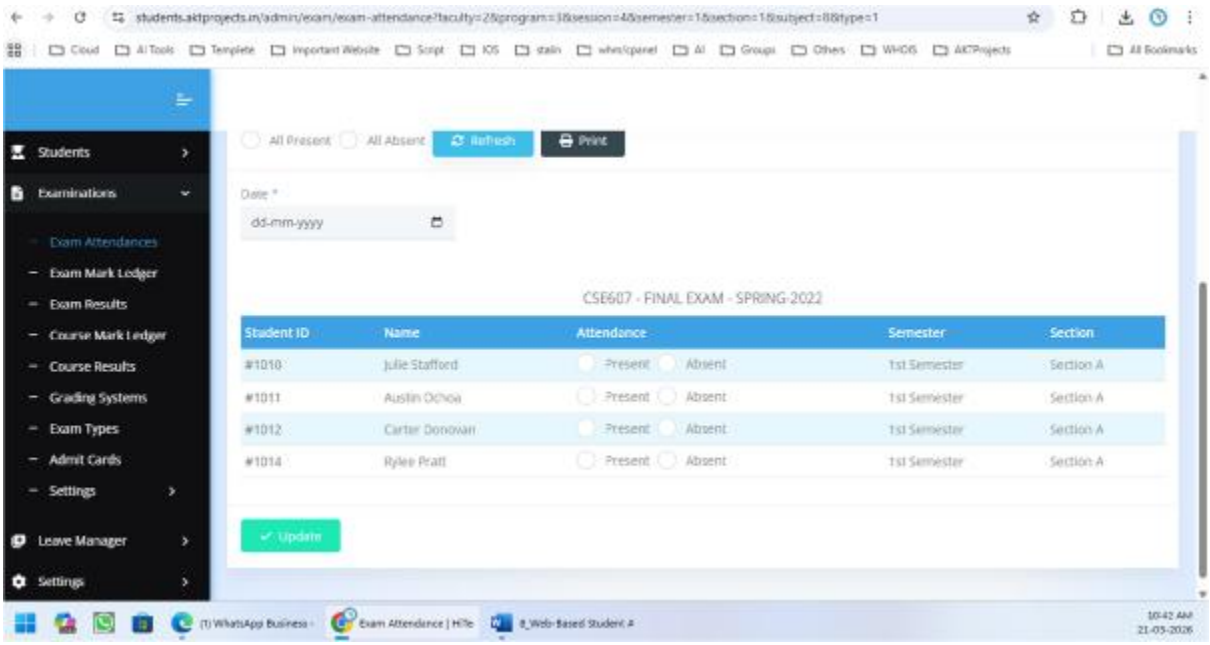
```
                    <p><mark class="text-primary">{{ __('field_description') }}:</mark> {!! $row->assignment->description !!}</p><hr/>
                  </div>
                </div>
              </div>
              <!-- Details View End -->
              </div>
            </div>
          </div>
          <div class="col-sm-4">
            <div class="card">
              @if(is_file('uploads/'.$path.'/'.$row->attach))
              <div class="card-header">
                {{ __('field_your_file') }} :
                <a href="{{ asset('uploads/'.$path.'/'.$row->attach) }}" class="btn btn-dark" download><i class="fas fa-download"></i> {{ __('btn_download') }}</a>
              </div>
              @endif

              @if($row->assignment->end_date >= date("Y-m-d"))
              <div class="card-block">
                <form class="needs-validation" novalidate action="{{ route($route.'.update', $row->id) }}" method="post" enctype="multipart/form-data">
                @csrf

                <div class="form-group">
                  <label for="attach">{{ __('field_upload_pdf') }}</label>
                  <input type="file" class="form-control" name="attach" id="attach" value="{{ old('attach') }}" accept=".pdf" required>

                  <div class="invalid-feedback">
```

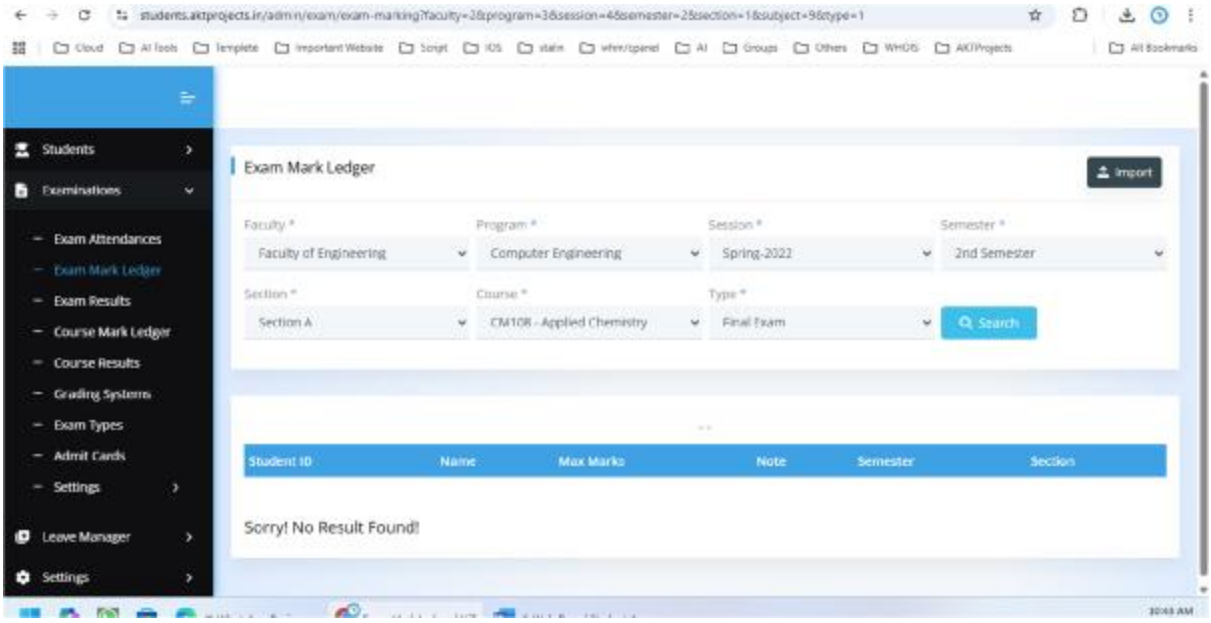
```
                    {{ __('required_field') }} {{ __('field_upload_pdf') }}
                  </div>
                </div>

                <button type="submit" class="btn btn-success"><i class="fas fa-upload"></i> {{ __('btn_upload') }}</button>
                </form>
              </div>
              @endif
            </div>
          </div>
        </div>
        <!-- [ Main Content ] end -->
    </div>
</div>
<!-- End Content-->

@endsection
```

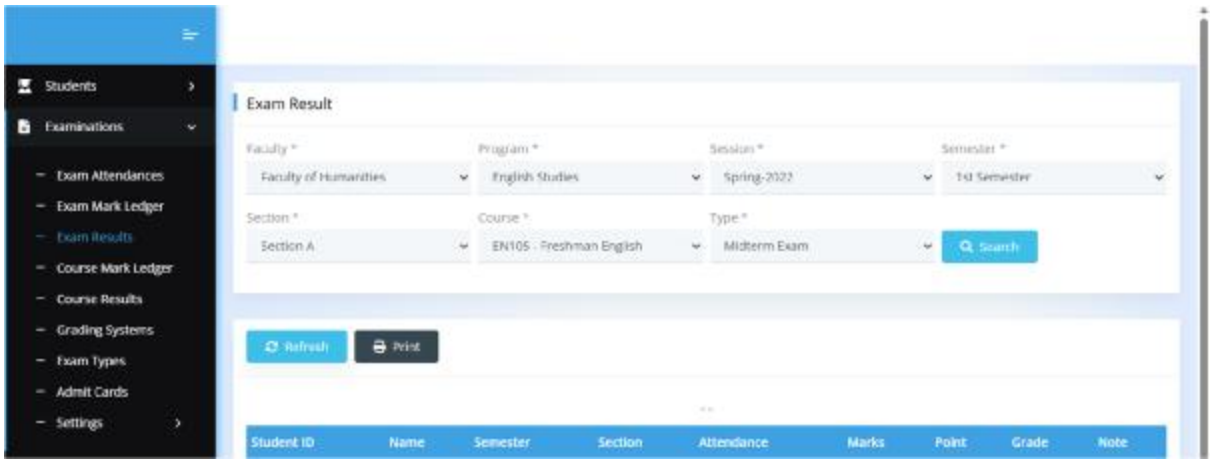
4.4. SCREEN SHOT

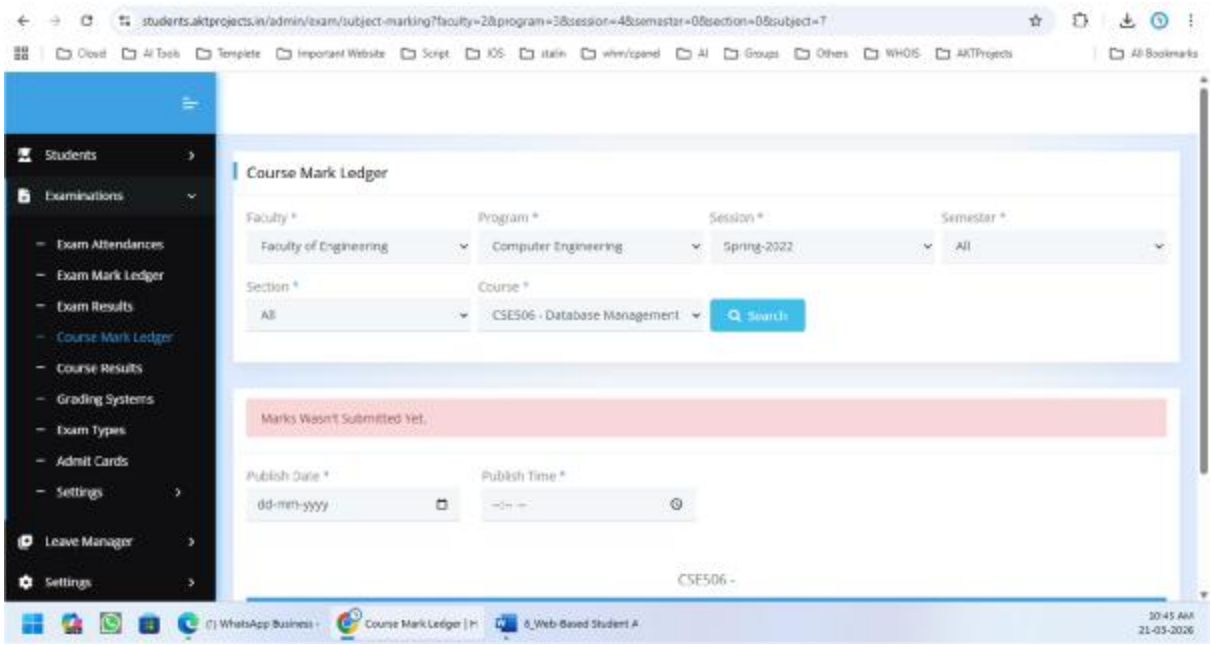


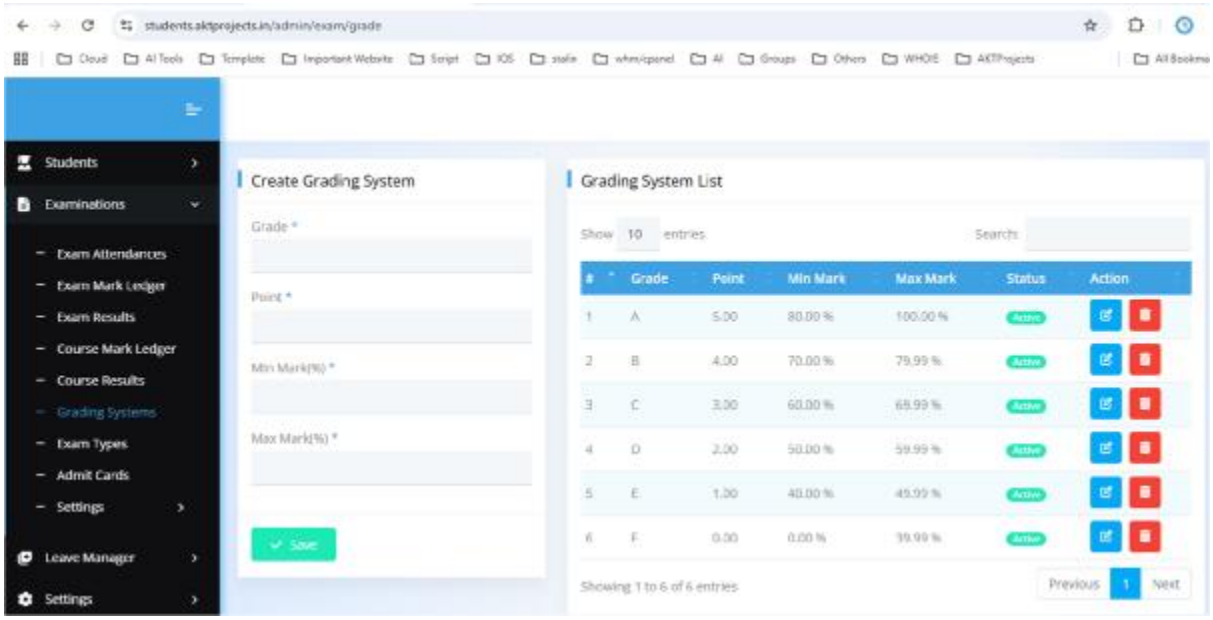


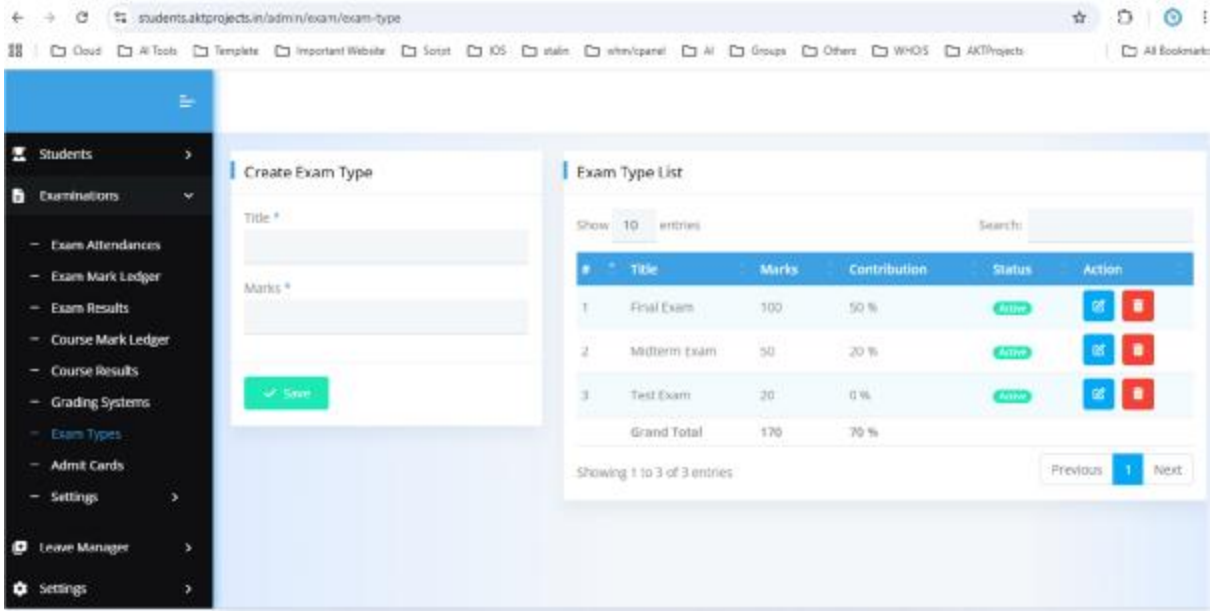

Students
Examinations
Exam Attendances
Exam Mark Ledger
Exam Results
Course Mark Ledger
Course Results
Grading Systems
Exam Types
Admit Cards
Settings
Leave Manager
Settings
All Present
All Absent
Refresh
Print
Date *
dd-mm-yyyy
CSE607 - FINAL EXAM - SPRING-2022
Student ID
Name
Attendance
Semester
Section
#1010
Julie Stafford
Present
Absent
1st Semester
Section A
#1011
Austin Ochoa
Present
Absent
1st Semester
Section A
#1012
Carter Donovan
Present
Absent
1st Semester
Section A
#1014
Rylee Pratt
Present
Absent
1st Semester
Section A
Update

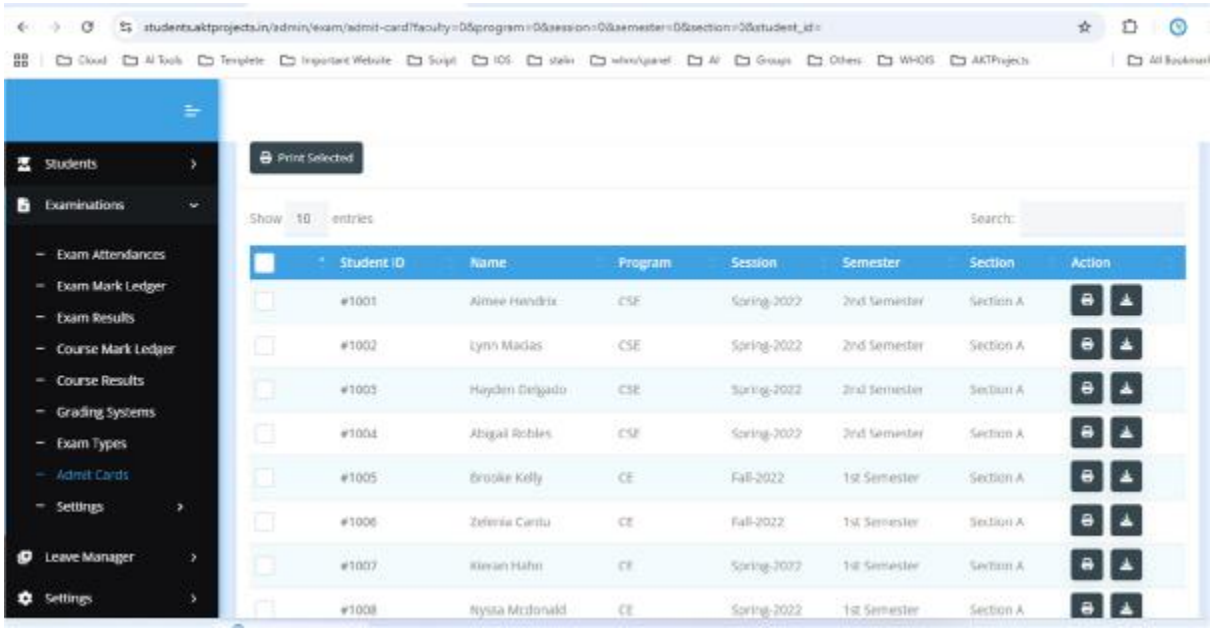

Students
Examinations
Exam Attendances
Exam Mark Ledger
Exam Results
Course Mark Ledger
Course Results
Grading Systems
Exam Types
Admit Cards
Settings
Leave Manager
Settings
Exam Mark Ledger
Import
Faculty *
Faculty of Engineering
Program *
Computer Engineering
Session *
Spring-2022
Semester *
2nd Semester
Section *
Section A
Course *
CM108 - Applied Chemistry
Type *
Final Exam
Search
Student ID
Name
Max Marks
Note
Semester
Section
Sorry! No Result Found!


Students
Examinations
Exam Attendances
Exam Mark Ledger
Exam Results
Course Mark Ledger
Course Results
Grading Systems
Exam Types
Admit Cards
Settings
Exam Result
Faculty *
Faculty of Humanities
Program *
English Studies
Session *
Spring-2022
Semester *
1st Semester
Section *
Section A
Course *
EN105 - Freshman English
Type *
Midterm Exam
Search
Refresh
Print
Student ID
Name
Semester
Section
Attendance
Marks
Point
Grade
Note


students.aktprojects.in/admin/exam/subject-marking?faculty=2&program=3&session=4&semester=0&section=0&subject=7
Students
Examinations
Exam Attendances
Exam Mark Ledger
Exam Results
Course Mark Ledger
Course Results
Grading Systems
Exam Types
Admit Cards
Settings
Leave Manager
Settings
Course Mark Ledger
Faculty *
Faculty of Engineering
Program *
Computer Engineering
Session *
Spring-2022
Semester *
All
Section *
All
Course *
CSE506 - Database Management
Search
Marks Wasn't Submitted Yet.
Publish Date *
dd-mm-yyyy
Publish Time *
CSE506 -
Course Mark Ledger | H
6_Web-Based Student A
10:45 AM
21-03-2026


students.aktprojects.in/admin/exam/grade
Students
Examinations
Exam Attendances
Exam Mark Ledger
Exam Results
Course Mark Ledger
Course Results
Grading Systems
Exam Types
Admit Cards
Settings
Leave Manager
Settings
Create Grading System
Grade *
Point *
Min Mark(%) *
Max Mark(%) *
Save
Grading System List
Show 10 entries
Search:
Grade Point Min Mark Max Mark Status Action
1 A 5.00 80.00 % 100.00 % Active
2 B 4.00 70.00 % 79.99 % Active
3 C 3.00 60.00 % 69.99 % Active
4 D 2.00 50.00 % 59.99 % Active
5 E 1.00 40.00 % 49.99 % Active
6 F 0.00 0.00 % 39.99 % Active
Showing 1 to 6 of 6 entries
Previous 1 Next


students.aktprojects.in/admin/exam/exam-type
Students
Examinations
Exam Attendances
Exam Mark Ledger
Exam Results
Course Mark Ledger
Course Results
Grading Systems
Exam Types
Admit Cards
Settings
Leave Manager
Settings
Create Exam Type
Title *
Marks *
Save
Exam Type List
Show 10 entries
Search:
Title Marks Contribution Status Action
1 Final Exam 100 50 % Active
2 Midterm Exam 50 20 % Active
3 Test Exam 20 0 % Active
Grand Total 170 70 %
Showing 1 to 3 of 3 entries
Previous 1 Next


Students
Examinations
Exam Attendances
Exam Mark Ledger
Exam Results
Course Mark Ledger
Course Results
Grading Systems
Exam Types
Admit Cards
Settings
Leave Manager
Settings
Print Selected
Show 10 entries
Search:
Student ID
Name
Program
Session
Semester
Section
Action
#1001 Aimee Hendrix CSE Spring-2022 2nd Semester Section A
#1002 Lynn Macias CSE Spring-2022 2nd Semester Section A
#1003 Hayden Delgado CSE Spring-2022 2nd Semester Section A
#1004 Abigail Robles CSE Spring-2022 2nd Semester Section A
#1005 Brooke Kelly CE Fall-2022 1st Semester Section A
#1006 Zelenia Cantu CE Fall-2022 1st Semester Section A
#1007 Kieran Hahn CE Spring-2022 1st Semester Section A
#1008 Nyssa Mcdonald CE Spring-2022 1st Semester Section A

5. CONCLUSION

The Web-Based Student Academic Performance Tracking System provides a modern and efficient solution for managing, monitoring, and analyzing student academic data within educational institutions. Traditional methods of maintaining records, such as manual registers or disconnected systems, are often inefficient, time-consuming, and prone to errors. This system successfully overcomes those limitations by introducing a centralized, automated, and user-friendly digital platform that enhances the overall academic management process.

One of the major advantages of the system is its ability to provide real-time access to academic information. Students can easily view their marks, attendance, and performance reports, while teachers can update records and track student progress without delays. This level of transparency improves communication between students, teachers, and administrators, creating a more collaborative and accountable educational environment. It also helps students become more aware of their performance and encourages them to take responsibility for their learning.

The system's analytical capabilities play a significant role in improving educational outcomes. By generating detailed reports and performance insights, it enables educators to identify students' strengths and weaknesses. Teachers can provide personalized support to students who are struggling, while also recognizing and motivating high-performing individuals. Institutions can use these insights to evaluate teaching strategies, improve curriculum design, and make informed academic decisions.

Another important strength of the system is its scalability and flexibility. Built using modern web technologies and a modular architecture, the system can be easily expanded to include additional features such as AI-based performance prediction, mobile applications, or integration with learning management systems. It can be deployed on local servers, VPS, or cloud platforms, making it suitable for institutions of different sizes.

Security and data integrity are also well addressed in the system. With features such as authentication, role-based access control, and data encryption, sensitive academic information is protected from unauthorized access. This ensures reliability and builds trust among users.

In conclusion, the Web-Based Student Academic Performance Tracking System is a powerful and practical tool that enhances efficiency, accuracy, and transparency in academic management. It supports data-driven decision-making, improves student engagement, and contributes to better educational performance. The system not only simplifies administrative tasks but also plays a crucial role in shaping a smarter and more effective learning environment.

REFERENCES

1. Software Engineering: A Practitioner's Approach – Roger S. Pressman, McGraw-Hill Education, 7th Edition, 2010.
2. Web Engineering: A Practitioner's Approach – Roger S. Pressman and David Lowe, McGraw-Hill, 2008.
3. Database System Concepts – Abraham Silberschatz, Henry F. Korth, S. Sudarshan, McGraw-Hill, 6th Edition, 2011.
4. PHP and MySQL Web Development – Luke Welling and Laura Thomson, Addison-Wesley, 5th Edition, 2016.
5. Laravel Framework – Official Documentation, https://laravel.com/docs
6. MySQL Database – Oracle Corporation, https://dev.mysql.com/doc/
7. Apache HTTP Server – Apache Software Foundation, https://httpd.apache.org/
8. Bootstrap Framework – Official Documentation, https://getbootstrap.com/
9. IEEE – Research papers on web-based systems and alumni platforms.
10. ACM – Digital Library for software engineering and web system research.
11. Google Analytics – https://analytics.google.com/
12. PayPal – https://www.paypal.com/
13. Stripe – https://stripe.com/
14. GitHub – https://github.com/
15. Modern Web Development – Multiple authors, technical publications on modern web applications.