COLLEGE EVENT MANAGEMENT SYSTEM
MINI PROJECT REPORT

Submitted by

THUSHEEL S [221501163]
VISHAL R [221501177]
DHANUS D [221501502]

In partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

RAJALAKSHMI ENGINEERING COLLEGE

(AUTONOMOUS) THANDALAM

CHENNAI-602105

RAJALAKSHMI
ENGINEERING COLLEGE

BONAFIDE CERTIFICATE

NAME ……..……..…

ACADEMIC YEAR…….……SEMESTER…….BRANCH……

UNIVERSITY REGISTER No. | |

Certified that this is the bonafide record of work done by the above students in the Mini Project titled "COLLEGE EVENT MANAGEMENT SYSTEM " during the year 2023 - 2024.

Signature of Faculty – in – Charge

Submitted for the Practical Examination held on _____

Internal Examiner **External Examiner**

ABSTRACT

The College Event Management System represents a dynamic solution to the multifaceted challenges of organizing and executing diverse events within a college setting. This comprehensive platform serves as a centralized hub, empowering administrators, faculty, and students alike to efficiently plan, coordinate, and manage a wide array of activities spanning from academic seminars and workshops to cultural festivals and sports tournaments. At its core, this mini-project endeavours to revolutionize the event management landscape by harnessing the power of technology to streamline processes, enhance communication, and optimize resource utilization..

Ultimately, the College Event Management System transcends traditional paradigms of event organization, ushering in a new era of efficiency, effectiveness, and excellence. By leveraging technology to optimize processes, amplify engagement, and enhance the overall event experience, this mini-project seeks to redefine the landscape of college event management, paving the way for a brighter, more vibrant future within the academic community.

LIST OF FIGURES

S NO.	TITLE	PAGE
3.2.1	ARCHITECTURE	18
3.3.1	ER DIAGRAM	21
5.1.1	PAGE	35
5.1.2	DASHBOARD	35

LIST OF ABBREVIATIONS

DBMS	Database Management System
SQL	Structured Query Language
PHP	Hypertext Preprocessor
HTML	Hyper Text Markup Language
CSS	Cascading Style Sheets
DOM	Document Object Model
HTTP	Hyper Text Transfer Protocol
ER	Entity Relationship
NF	Normal Form

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO
1	**INTRODUCTION**	1
	1.1 INTRODUCTION	1
	1.2 OBJECTIVES	2
	1.3 MODULES	3
2	**SURVEY OF TECHNOLOGIES**	10
	2.1 SOFTWARE DESCRIPTION	10
	2.2 LANGUAGES	14
3	**REQUIREMENTS AND ANALYSIS**	15
	3.1 REQUIREMENT SPECIFICATION	15
	3.2 HARDWARE AND SOFTWARE REQUIREMENTS	18
	3.3 ARCHITECTURE DIAGRAM	19
	3.4 ER DIAGRAM	21
	3.5 NORMALIZATION	22
4	**PROGRAM CODE**	25
	4.1 SAMPLE CODE	25
5	**RESULTS AND DISCUSSION**	33
	5.1.1 LOGIN PAGE	35
	5.1.2 DASHBOARD	35
6	**CONCLUSION**	36
7	**REFERENCES**	37

CHAPTER 1

INTRODUCTION

1.1. INTRODUCTION

The College Cultural Event Management System is a comprehensive web-based solution aimed at enhancing the efficiency and effectiveness of organizing, managing, and participating in cultural events within a college environment.

Cultural events are an integral part of college life, providing students with opportunities to showcase their talents, engage in creative pursuits, and foster a sense of community. However, managing these events involves numerous logistical challenges, including coordinating event details, handling participant registrations, and maintaining accurate records.

This project not only aims to improve the operational efficiency of cultural event management but also to provide a scalable and secure solution that can be adapted to the unique needs of any educational institution. Through this system, colleges can ensure that their cultural events are organized more effectively, allowing for greater student participation and engagement.

1.2. OBJECTIVES

The primary objective of the College Cultural Event Management System is to streamline the process of organizing and managing cultural events. This includes:

1. **Efficient Event Creation and Management**: Enabling administrators to easily create, update, and delete events, along with detailed information such as event name, date, time, venue, description, and participants.
2. **User-Friendly Registration**: Providing students with an intuitive interface to view event details and register online, thereby enhancing student participation.
3. **Accurate Record-Keeping**: Maintaining a centralized database of all events and registrations to ensure data accuracy and ease of access.
4. **Data Export and Reporting**: Allowing administrators to export event and participant data for reporting and analysis purposes.

1.3. MODULES

1. User Authentication Module

- **Purpose**: To ensure secure access to the system.
- **Features**:
  - Login and logout functionality for administrators.
  - Password encryption and validation.
  - Session management to prevent unauthorized access.

The purpose of implementing secure access to the college event management system is to ensure that only authorized users can perform administrative tasks, thereby protecting sensitive data and maintaining the integrity of the system.

Key features to achieve this include login and logout functionality for administrators, which provides a controlled entry and exit point to the system, ensuring that only verified users can access administrative functions. Password encryption and validation are crucial for safeguarding user credentials, as encrypted passwords make it significantly harder for attackers to decipher and misuse them.

Additionally, robust session management mechanisms are employed to prevent unauthorized access; by tracking active sessions and automatically logging out users after a period of inactivity or when they manually log out, the system mitigates the risk of unauthorized access. Together, these features form a comprehensive security framework that upholds the confidentiality, integrity, and availability of the system.

2. **Event Management Module**

- **Purpose**: To facilitate the creation, updating, and deletion of events.
- **Features**:
    - Form for adding new events with fields for event name, date, time, venue, description, and participants.
    - Edit functionality to update existing event details.
    - Delete functionality to remove events from the system.
    - Validation to ensure all necessary event details is provided.

3. **Student Registration Module**

- **Purpose**: To allow students to register for events.
- **Features**:
    - List view of all upcoming events.
    - Registration form for students to sign up for events.
    - Confirmation messages to acknowledge successful registration.
    - Cancellation option for students to withdraw from events.

4. **Database Management Module**

- **Purpose**: To handle all data storage and retrieval operations.
- **Features**:
    - Integration with MySQL to store event and participant details.
    - Queries for inserting, updating, deleting, and fetching data.

- Backup and restore functionality to protect data integrity.

5. **Data Export Module**

- **Purpose**: To provide an option to export event and participant data for reporting purposes.
- **Features**:
    - Export data to Excel format using PhpSpreadsheet or similar libraries.
    - Option to download participant lists and event details.
    - Filters to export data for specific events or date ranges.

6. **User Interface Module**

- **Purpose**: To provide a responsive and interactive interface for both administrators and students.
- **Features**:
    - User-friendly forms and tables for data entry and display.
    - CSS and JavaScript for styling and interactivity.
    - Responsive design to ensure compatibility with various devices.

Reporting Module

- **Purpose**: To generate reports on event participation and other metrics.
- **Features**:
    - Generate summary reports of all events.

- Detailed reports on individual events, including participant statistics.
- Graphical representation of data for better visualization.

7. **Admin Dashboard Module**

- **Purpose**: To provide a centralized control panel for administrators.
- **Features**:
  - Overview of all events and registrations.
  - Quick access to event management and data export options.
  - Activity logs to track changes and updates.

CHAPTER 2

SURVEY OF TECHNOLOGIES

2.1 SOFTWARE DESCRIPTION

The College Cultural Event Management System is a web-based application designed to streamline the organization and management of cultural events within a college or university. This system provides a comprehensive solution for event creation, student registration, data management, and reporting, ensuring a seamless experience for both administrators and participants.

2.1.1 Functional Requirements

Functional requirements are detailed statements that specify the functions and behaviors a system must have to meet the needs of its users and stakeholders. These requirements describe the tasks the system should perform, the operations it must support, and the ways users will interact with it. Here is a basic explanation of functional requirements using an example of a cultural event management system:

- **User Authentication:**
  - Secure login and logout for administrators.
  - Password encryption and validation.
- **Event Management:**
  - Add, update, and delete events.
  - Validate event details to ensure completeness.
- **Student Registration:**
  - Display list of events for students to view.

- Register students for selected events.
- Provide confirmation and cancellation options.

➢ **Data Export:**

- Export event and participant data to Excel.
- Filter data based on specific criteria.

2.1.2Non-Functional Requirements

Non-functional requirements specify the quality attributes, performance, and constraints of a system. They define how the system performs and operates rather than what it should do. Following subtopic are NFR of this project:

➢ **Performance:**

- Efficient handling of multiple user requests.
- Quick data retrieval and processing.

➢ **Scalability:**

- Ability to handle an increasing number of events and participants.

➢ **Security:**

- Protection against SQL injection and cross-site scripting (XSS).
- Secure data transmission using HTTPS.

➢ **Usability:**

- User-friendly interface with intuitive navigation.
- Responsive design for compatibility with various devices.

2.1.3Database Design

- **Tables:**
    - **Events:** Stores event details (event ID, name, date, time, venue, description, participants).
    - **Users:** Stores user credentials and roles.
    - **Registrations:** Tracks student registrations for events.
- **Relationships:**
    - One-to-many relationship between events and registrations.
    - Users can have roles (e.g., administrator).

2.1.4User Interface Design

- **Admin Dashboard:**
    - Overview of events and registrations.
    - Links to add, update, and delete events.
    - Export data and generate reports.
- **Event List:**
    - Table displaying all events with options to edit or delete.
    - Registration form for students.
- **Registration Form:**
    - Form fields for entering student details and selecting events.
    - Confirmation message on successful registration.

2.1 LANGUAGES

- **Frontend:**
  - HTML5, CSS3: For structuring and styling the web pages.
  - JavaScript: For interactive features and dynamic content.
- **Backend:**
  - PHP: For server-side scripting and handling business logic.
  - MySQL: For database management and storage of event-related data.
- **Libraries and Frameworks:**
  - PhpSpreadsheet: For exporting data to Excel.
- **Tools:**
  - XAMPP/WAMP: For local development environment setup.
  - PhpMyAdmin: For database management.
- **IDE:** Visual Studio Code or any preferred IDE for development.

CHAPTER 3

REQUIREMENTS AND ANALYSIS

3.1 REQUIREMENT SPECIFICATION

Requirements refer to the conditions or capabilities that must be met or possessed by a system, component, or software product to satisfy a contract, standard, specification, or other formally imposed documents. Requirements are the foundation for the development of any project and are critical for ensuring that the final product meets the needs and expectations of stakeholders. Requirements can be classified into different types:

1. User Requirements

1. User Authentication

- Login/Logout: Secure authentication for both administrators and students.
- Role-Based Access: Different functionalities accessible based on user roles.

2. Event Management

- Create Events: Administrators can add new events with details like event name, date, venue, description, and participant limit.
- Update Events: Administrators can modify existing event details.
- Delete Events: Administrators can remove events from the system.
- View Events: Both administrators and students can view a list of all events with relevant details.

3. Student Registration

- Register for Events: Students can register for events and receive confirmation.

- Cancel Registration: Students can cancel their registration for events.
- View Registered Events: Students can see the events they are registered for.

4. Data Export

- **Export to Excel:** Administrators can export event details and participant lists to Excel files for record-keeping and reporting.

System Requirements

2. Functional Requirements

- **Database Management**
  - Store user information, event details, and registration data securely.
  - Provide efficient data retrieval for display and reporting.
- **Event CRUD Operations**
  - Enable administrators to create, read, update, and delete event records in the database.
  - Ensure data integrity and validation during CRUD operations.
- **Registration Management**
  - Allow students to register and cancel registration for events.
  - Update the database with registration changes in real-time.

Non-Functional Requirements

- **Performance**
  - The system should handle multiple concurrent users efficiently.
  - Response times for data retrieval and updates should be minimal.
- **Scalability**
  - The system should be able to accommodate an increasing number of users and events.
  - Design the database and application logic to handle growth

without significant performance degradation.

- **Security**
  - Implement strong password policies and encryption for sensitive data.
  - Protect against common vulnerabilities such as SQL injection and cross-site scripting (XSS).

3.2 HARDWARE AND SOFTWARE REQUIREMENTS

Hardware Requirements

Hardware requirements specify the physical components needed to support the system's development, deployment, and operation. For a college event management system, the hardware requirements can be divided into server-side and client-side components

- **Requirements:**
    - Processor: Intel Core i5 or equivalent
    - RAM: 8 GB or higher
    - Storage: 100 GB HDD or SSD
    - Network: Reliable network connection with at least 100 Mbps speed

Software Requirements

- **Server-Side:**
    - Operating System: Windows Server 2012 or later, or any Linux distribution such as Ubuntu 18.04 or later
    - Web Server: Apache 2.4 or Nginx
    - Database Server: MySQL 5.7 or later

3.2 ARCHITECTURE DIAGRAM

An architectural diagram is a visual representation that outlines the structure and organization of a system, showing the relationships and interactions between its components. It provides a high-level overview of how the system is designed, focusing on components, modules, and their interactions. Architectural diagrams help stakeholders understand the system's layout and facilitate communication among developers, designers, and other project participants.

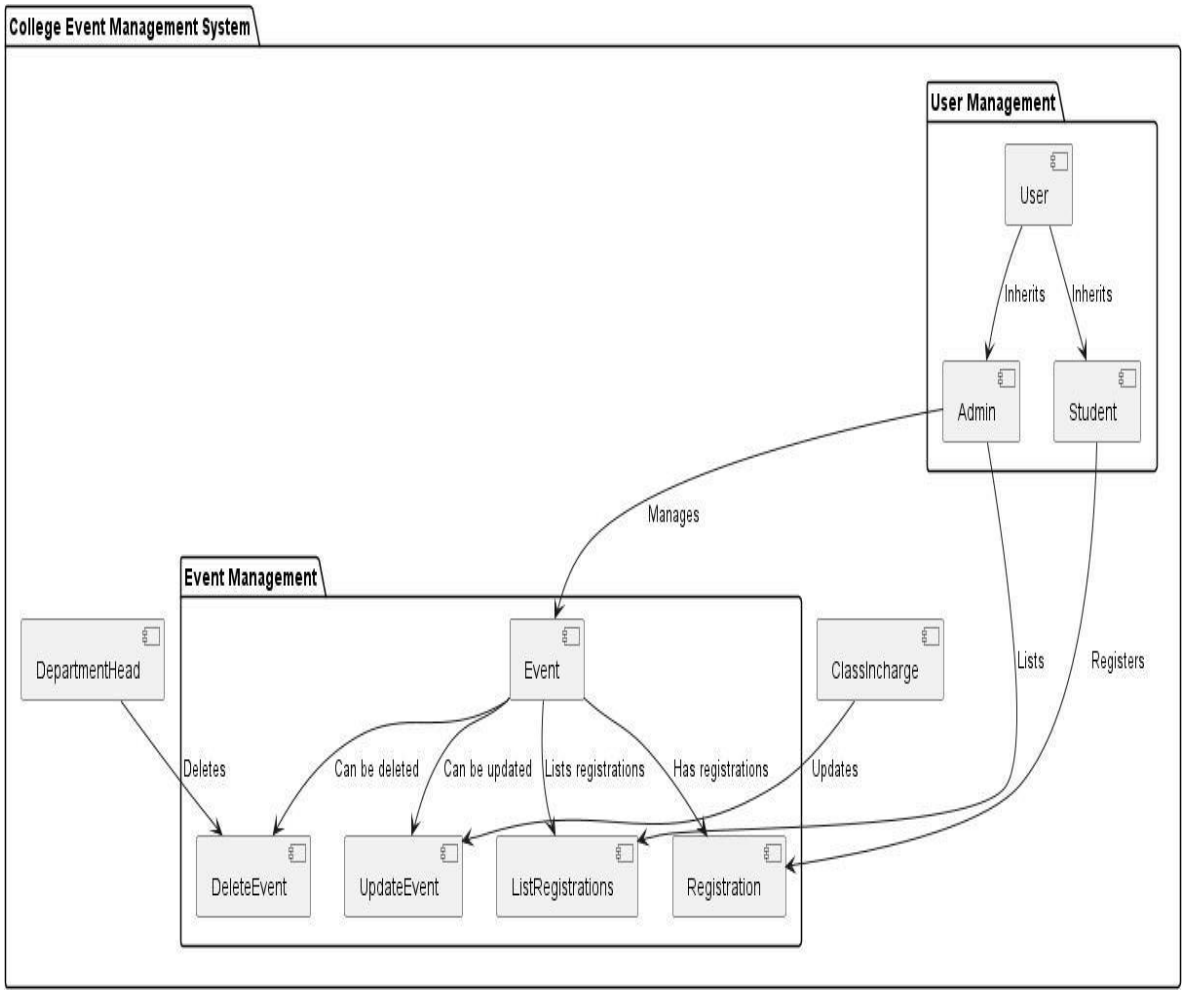


FIGURE 3.2.1 ARCHITECTURE DIAGRAM

Event Management Component:

1. **Entities:**

    - **DepartmentHead**: Represents department heads or administrators who oversee events. They have the authority to manage events.
    - **Event:** Represents various events organized by the college. Events have attributes such as EventName, EventDate, and EventLocation. The EventID serves as the primary key.
    - **Registration:** Represents user registrations for events. It includes information like RegistrationDate. The RegistrationID is the primary key.

2. **Actions and Relationships:**

    - **DeleteEvent:** Indicates that an event can be deleted by the DepartmentHead.
    - **Can be deleted:** Connects the DeleteEvent action to the Event entity.
    - **Can be updated:** Indicates that an event's details can be modified.
    - **Lists registrations:** Represents events that display their registrations.
    - **Has registrations:** Connects the Event entity to the Registration entity.

3.3 ER DIAGRAM

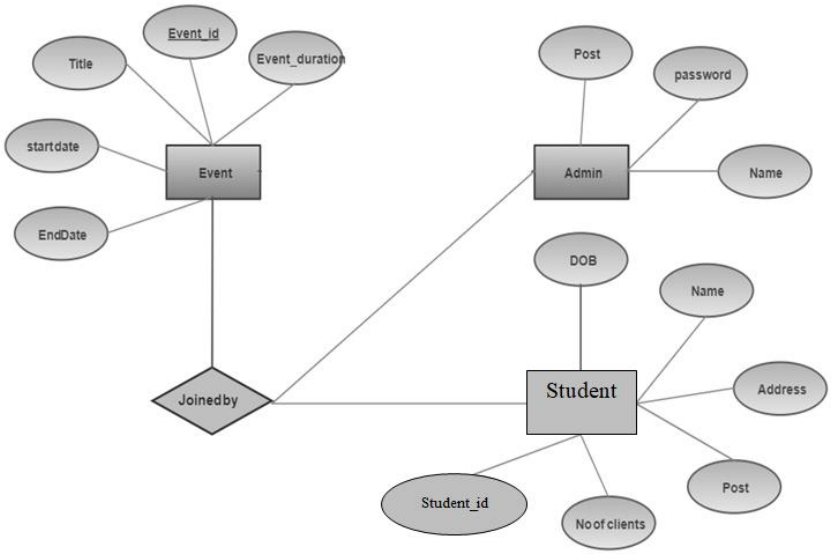


FIGURE 3.3.1 ER DIAGRAM

The provided diagram is an Entity-Relationship (ER) diagram illustrating the relationships between entities in a student-event management system. It includes three main entities: `Event`, `Student`, and `Admin`. The `Event` entity has attributes such as `Title`, `startdate`, `EndDate`, `Event_id`, and `Event_duration`. The `Student` entity has attributes like `Student_id`, `DOB`, `Name`, `Address`, `Post`, and `Noofclients`. The `Admin` entity is characterized by `Post`, `password`, and `Name`. The relationship `Joinedby` connects `Event` and `Student`, indicating that students participate in events. Each entity and attribute is represented graphically, showing their interconnections and relationships.

3.5 NORMALIZATION

We identify the main entities and their attributes based on the functions of the college event management system:

1. **User**
    - UserID (Primary Key)
    - UserName
    - Password
    - UserType
2. **Event**
    - EventID (Primary Key)
    - EventName
    - EventDate
    - EventLocation
3. **Registration**
    - RegistrationID (Primary Key)
    - UserID (Foreign Key referencing User.UserID)
    - EventID (Foreign Key referencing Event.EventID)
    - RegistrationDate
4. **UpdateEvent**
    - UpdateEventID (Primary Key)
    - EventID (Foreign Key referencing Event.EventID)
    - UpdatedDate
5. **DeleteEvent**
    - DeleteEventID (Primary Key)
    - EventID (Foreign Key referencing Event.EventID)
    - DeletedDate

6. **ListRegistrations**
    - ListID (Primary Key)
    - EventID (Foreign Key referencing Event.EventID)
    - UserID (Foreign Key referencing User.UserID)
    - ListDate

Step 2: Apply First Normal Form (1NF)

In 1NF, we ensure that each table has atomic values and no repeating groups or arrays. This is usually achieved by breaking down composite attributes into separate columns.

Step 3: Apply Second Normal Form (2NF)

In 2NF, we ensure that all attributes are fully functionally dependent on the entire primary key. If any attribute is functionally dependent on only part of the primary key, we create separate tables to resolve this.

- User and Event tables are already in 2NF as all attributes depend on their respective primary keys.
- Registration, UpdateEvent, DeleteEvent, and ListRegistrations tables each have a composite primary key, so we need to check for partial dependencies.

Step 4: Apply Third Normal Form (3NF)

In 3NF, we eliminate transitive dependencies. If an attribute depends on another non-key attribute, it should be moved to its own table.

- **Registration table**: UserID and EventID depend on RegistrationID (Primary Key). No transitive dependencies.
- **UpdateEvent table:** EventID depends on UpdateEventID (Primary Key). No transitive dependencies.
- **DeleteEvent table:** EventID depends on DeleteEventID (Primary Key). No transitive dependencies.
- **ListRegistrations table:** EventID and UserID depend on ListID (Primary Key). No transitive dependencies.

Final Normalized Schema

Based on the normalization process, we have a set of tables that are in 3NF, ensuring data integrity and minimizing redundancy:

- User
- Event
- Registration
- UpdateEvent
- DeleteEvent
- ListRegistrations

CHAPTER 4

PROGRAM CODE

4.1 SAMPLE CODE

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Admin Page</title>
<style>
body {
  font-family: Arial, sans-serif;
}

.container {
  max-width:
  800px; margin: 0
  auto; padding:
  20px;
  background-color:
  #f5f5f5; border-
  radius: 10px;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

.header {
```

```
    text-align: center;
    margin-bottom:
    20px;
}

.navbar {
    display:
    flex;
    justify-content: space-
    between; background-
    color: #333; border-
    radius: 5px;
    padding: 10px;
}

.navbar a {
    color: #fff;
    text-decoration:
    none; padding:
    8px 15px;
    border-radius:
    5px;
}

.navbar a:hover {
    background-color:
    #555;
}
```

```
.content {
   background-color:
   #fff; padding: 20px;
   border-radius: 5px;
}
.footer {
   text-align:
   center;
   margin-top:
   20px;     color:
   #777;
}
li{
   list-style: none;
    padding-top: 10px; font-size: 20px;
}
i{
   padding-right:
   10px; font-size:
   30px;
}
a{
   color: black;
}
</style>
</head>
```

```
<body>
<div class="container">
  <div class="header">
    <h1>Welcome  Admin</h1>
  </div>
  <div class="navbar">
  <a href="#" class="menu-bar" onclick="showhome()">Home</a>
  <a href="#" class="menu-bar" onclick="showevent()">Add Event</a>
  <a href="#" class="menu-bar" onclick="showaddedevent()">Added
  Events</a>
  <a href="#" class="menu-bar" onclick="showstdregister()">Register Student
  List</a>
  <a href="#" class="menu-bar" onclick="showcontact()">contact us</a>
  <a href="adminlogin.php">logout</a>

  </div>
  <div id="home">
    <h2>Home</h2>
    <p>The College Event Management System is a comprehensive platform
designed to streamline and enhance the planning, organization, and execution
of various events within a college or university setting. It encompasses a range
of features and functionalities aimed at Simplifying event management tasks,
improving communication, and ensuring successful events. .</p>

    <div class="footer">
    <p>&copy; SK College Event Management System</p>
  </div>
  </div
  <div id="addevent">
    <h2>Add Events</h2>
    <ul class="accordion">
```

```
<input type="radio" name="accordion" id="sixth">
<label for="sixth">Cultural Events</label>
<div class="content">
<p>These include music concerts, dance performances, drama plays, fashion shows, and talent competitions</p>
<img src="adminimage/cultural.jpg">
<a href="addedculturalsevent.php">Added cultural events</a>
</div>
</li>
<li>
<input type="radio" name="accordion" id="seventh">
<label for="seventh">Sports Events</label>
<div class="content">
<p>Colleges often organize sports tournaments, athletic meets, and intercollegiate sports competitions in various disciplines like football, basketball, cricket, athletics, etc.</p>
<img src="adminimage/sports.jpg">
<a href="addedsportevent.php">Added sports event</a>
</div>
</li>
<li>
<input type="radio" name="accordion" id="eighth">
<label for="eighth">Academic Events</label>
<div class="content">
<p>These include seminars, workshops, conferences, symposiums, and guest lectures by experts in different fields.</p>
<img src="adminimage/academic.jpg">
<a href="addedacademicevent.php">Added academic event</a>
</div>
</li>
```

```
        <li>
          <input type="radio" name="accordion" id="ninth">
          <label for="ninth">Alumni Events</label>
          <div class="content">
            <p>Colleges may organize alumni reunions, networking events, and alumni mentorship programs.</p>
            <img src="adminimage/alumini.jpg">
            <a href="addedalumnievent.php">Added alumni event</a>
          </div>
<li>
<input type="radio" name="accordion" id="tenth">
          <label for="tenth">Placement and Training Events</label>
          <div class="content">
            <p>These include job fairs, campus recruitment drives, career counseling sessions, and resume-building workshops.</p>
            <img src="adminimage/placement.jpg">
            <a href="addedplacementevent.php">Added Placement and Training event</a>
          </div>
        </li></ul>
    </div>
    <div id="studentregisterevent">
      <h2>Student Register Events</h2>
      <ul class="accordion">
        <li><a href="academicregisterstudent.php">Registered list of academic events</a>
          </div>
        </li>
        <li>
          <input type="radio" name="accordion" id="Fourteen">
          <label for="Fourteen">Alumni Events</label>
```

```
<div class="content">
<p>Colleges may organize alumni reunions, networking events, and alumni mentorship programs.</p>
<img src="adminimage/alumini.jpg">
<a href="alumniregisterstudent.php">Registered list of alumni events</a>
</div>
</li>
<li>
<input type="radio" name="accordion" id=" Fifteen">
<label for=" Fifteen">Placement and Training Events</label>
<div class="content">
<p>These include job fairs, campus recruitment drives, career counseling sessions, and resume-building workshops.</p>
<img src="adminimage/placement.jpg">
<a href="placementregisterstudent.php">Registered list of Placement events</a>
</div>
</li>
a{
color: white;
text-align:
center; cursor:
pointer;
}
#addevent{
display:
none;
}
```

```
    #addedevent{
       display:
       none;
    }
    #studentregisterev
       ent{ display:
       none;
    }
    #contactus{
       display:
       none;
    }
    img{
       width: 35%;
    }
 #addevent{

padding: 10px
12%; text-align:
center; font-size:
35px; font-weight:
600;
 }

 #addedevent{
    padding: 10px
    12%; text-align
```

CHAPTER 5

RESULTS AND DISCUSSION

Results:

1. **Functionalities Implemented:**

   - User registration and login functionality are implemented.
   - Event creation, update, deletion, and listing functionalities are available.
   - User registration data is stored in the database.
   - Event data is stored and managed in the database.
   - Registration for events is recorded in the database.

2. **Database Interaction:**

   - The system connects to a MySQL database using PHP's MySQLi extension.
   - SQL queries are used to insert, update, delete, and retrieve data from the database.
   - User authentication and event management operations are performed securely through prepared statements to prevent SQL injection.

3. **User Interface:**

   - Basic HTML templates are used for the user interface, including header and footer sections for navigation.
   - Registration and login forms are provided for users.
   - Event listing is displayed in a structured format.

Discussion:

4. **System Architecture:**
    - The project follows a basic MVC (Model-View-Controller) architecture, where models represent data structures, controllers handle business logic and database operations, and views manage user interface presentation.
    - However, in the provided single-page code, the separation of concerns is limited due to the code's compact nature. In a real-world application, it's recommended to separate concerns more clearly for better maintainability.
5. **Database Design and Normalization:**
    - The project demonstrates basic database design principles with tables for users, events, and registrations.
    - Further normalization could be applied to optimize the database structure, such as breaking down user roles into separate tables or ensuring data integrity with foreign key constraints.
6. **Security Considerations:**
    - The project uses prepared statements to mitigate SQL injection attacks.
    - However, additional security measures such as password hashing and session management could enhance overall system security.

OUTPUT SCREENSHOT:

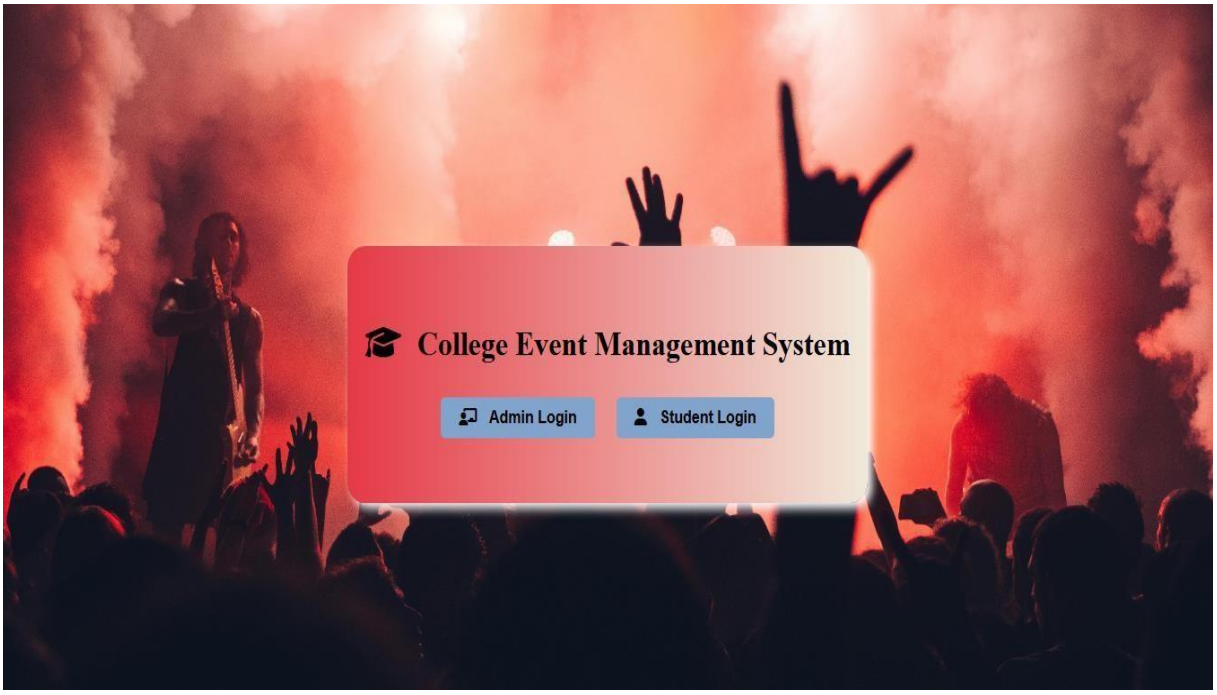
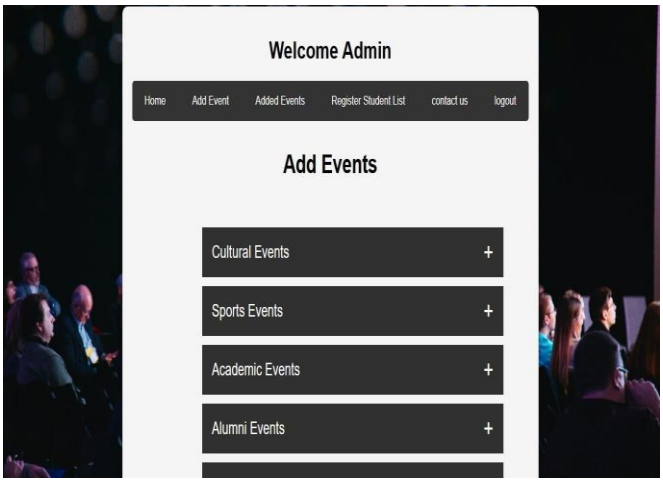


FIGURE 5.1.1 LOGIN PAGE



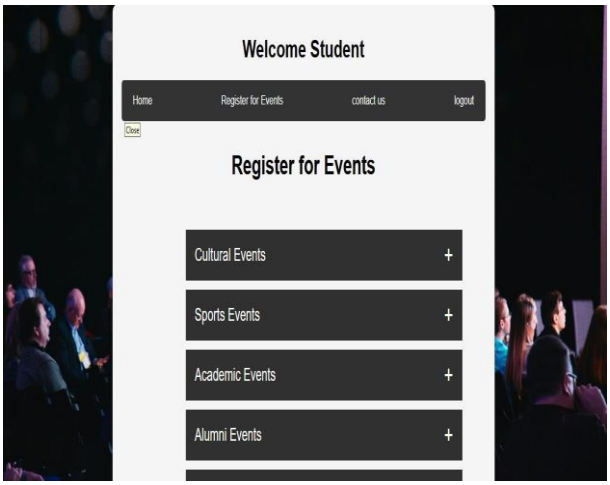


FIGURE 5.1.2 DASHBOARD

CHAPTER 6

CONCLUSION

The college event management system project, developed using PHP, represents a functional foundation for handling event-related activities within a college setting. Through features like user registration, event creation, updating, and deletion, as well as event registration for users, the system demonstrates core functionalities crucial for managing college events efficiently.

The project's adherence to security practices, including the use of prepared statements to prevent SQL injection, reflects a commitment to data integrity and user protection. However, for real-world deployment, considerations such as password hashing, session management, and enhanced input validation would further bolster the system's security posture.

The project's basic MVC architecture, albeit in a single-page format, provides a structured approach to code organization, promoting better maintainability and scalability. Moving forward, enhancements in user experience, such as improved event listing and registration processes, along with advanced features like role-based access control and event categorization, could elevate the system's functionality and user satisfaction. Overall, while the project serves as a functional prototype, continuous refinement and expansion are essential to meet the evolving needs of a comprehensive college event management solution.

CHAPTER 7

REFERENCES

1. Welling, Luke, and Laura Thomson. "PHP and MySQL Web Development." Addison- Wesley Professional, 2016.
2. Ullman, Larry. "PHP and MySQL for Dynamic Web Sites." Peachpit Press, 2017.
3. Freeman, Eric. "Head First Design Patterns." O'Reilly Media, 2004.
4. Fowler, Martin. "Patterns of Enterprise Application Architecture." Addison-Wesley Professional, 2002.
5. Schach, Stephen R. "Object-Oriented and Classical Software Engineering." McGraw- Hill Education, 2016.
6. IEEE Computer Society. "IEEE Xplore Digital Library." https://ieeexplore.ieee.org/.