

Project Report on
Event Management System

Submitted by
Shirsendu Roy and Sarnaavho Pal
Under the supervision and guidance of
Dr. Deepsubhra Guha Roy and
Dr. Bipasha Guha Roy

In the partial fulfilment of requirements for the award of Degree in
Bachelors of Technology
Batch 2022 - 2026
Submitted to the



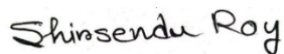
Institute of Engineering and Management
Salt Lake, Kolkata – 700064

DECLARATION

We hereby declare that the work recorded in this project report entitled "Event Management System" in partial fulfilment for the requirements for the award of Degree in Bachelors of Technology from Institute of Engineering and Management, Salt Lake, Kolkata, is a faithful and bona fide work carried out under the supervision and guidance of Dr. Deepsubhra Guha Roy and Dr. Bipasha Guha Roy.

The results of this investigation reported in this project have so far not been reported for any other Degree / Diploma or other technical forum.

The assistance and help received during the course of the investigation have been duly acknowledged.



STUDENT SIGNATURE

Name: Shirsendu Roy

Enrolment no.: 12022002016006



STUDENT SIGNATURE

Name: Sarnaavho Pal

Enrolment no.: 12022002016036

ABSTRACT

As a robust web application for managing different kinds of events, ranging from corporate conferences to social gatherings, the Event Management System (EMS) creates a dynamic environment with the help of the technology stack of HTML, CSS, Bootstrap, JavaScript, MySQL, and PHP to improve the experience of not only the organizers but the attendees too.

The frontend will be developed in the use of HTML and CSS. It will apply Bootstrap, which allows it to give a responsive layout and current styling that makes it versatile in use on various devices and screen sizes. Its backend will make use of PHP, which is used for the processing of requests from users and managing data in interaction with a MySQL database that keeps information safe about events, user profiles, registration details, and feedback.

The EMS boasts key functionalities in the form of event creation and editing tools, a registration module, real-time notifications, attendee management, and automated email reminders. Organizers have access to analytics and reports about participation in events, user permission management, and interaction with attendees through personalized dashboards. The system also accommodates ticketing, secure payment integration, and feedback collection towards improving future events.

The EMS has been so designed to overcome problems that often affect event management, including data coordination, participant engagement, and operational efficiency. Modular design and full functionalities make it an indispensable tool for event organizers who have all elements of event planning and execution streamlined, visually appealing, and optimized to succeed.

ACKNOWLEDGEMENT

We would like to express our special thanks of gratitude to “Dr. Deepsubhra Guha Roy” and “Dr. Bipasha Guha Roy”, for his able guidance and support in completing our project. We would also extend our gratitude to the HOD “Prof. Amartya Mukherjee” for providing us with all the facilities that was required.

TABLE OF CONTENTS

S.no.	Title	Page
1	Declaration	2
2	Abstract	3
3	Acknowledgement	3
4	Introduction	5
5	Litreture Review	6
6	System Description	7
7	Theoretical analysis	10
8	Methodology	14
9	Conclusion	26

INTRODUCTION

Overview

Event management entails elaborate planning and coordination of several aspects, such as scheduling, registrations, attendee engagement, and submitting comments on the events. Traditionally it involves extensive manual labor and communication, which may then lead to inefficiencies and challenges—especially as the scale and complexity of events increase. To help curb such issues, we came up with an EMS using modern web technologies for the automation and staging of the event planning process.

The EMS is coded using HTML, CSS, Bootstrap, JavaScript, MySQL, and PHP. This system provides intuitive and responsive interfaces with compatibility across the vast range of devices, thus ensuring ease of use and access for all parties. The server-side application with PHP supports dynamic data processing and easy interaction with a MySQL database to securely store all event and user information.

In the EMS, the organizers will facilitate events much more efficiently by taking care of all the registrations, sending out notifications, and attendance tracking. Additionally, it incorporates all the great features from an event dashboard right up to real-time updates, attendee management, automated emails, and ticketing into a single interface. The responsive grid system assures that the application looks good and its consistency is maintained, and JavaScript provides real-time feedback along with smooth client-side interactions.

The EMS shall design a scalable, reliable, and user-centric event management solution which would be imbued with all the tools that would be required in order to tackle the biggest of events through minimal manual intervention without increasing workload and to enhance the outcome of events. This EMS shall give the organizers an opportunity to care about delivering interesting and memorable events more than much focusing on complexities of operations by bringing all the features on one system in streamlined and accessible ways.

LITRATURE REVIEW

There is a considerable evolution in event management in recent times, where digital platforms have really disturbed the event planning, organization, and execution. Traditionally event management relied on manual operations and face-to-face contacts, which posed enormous challenges to coordinators especially in coordinating details, registrations, and communicating with participants. They have today grown more sophisticated and are growing bigger, better, more complicated, and finally demanding workflows and automated systems for logistics and attendee engagement.

An insight from the literature study on event management systems provides a review of these core problems faced by organizers which include management of event data, communication management with attendees, and overall event experience improvement. Manual event planning has an associated cost in terms of mistakes that are possible due to human error and the labor invested. The increasingly centralized and automated systems have given rise to digital event management solutions which span an entire range of features from scheduling, ticketing, notifications, and post-event feedback collection.

Exponential growth in web technologies such as HTML, CSS, JavaScript, and PHP has opened up avenues for developing easy-to-use and responsive applications. On research it is shown that platforms built with these web technologies coupled with responsive web design frameworks such as Bootstrap facilitate user interfaces that are easily accessible to all and have uniformity thus leading to user satisfaction. For example, with a database like MySQL being integrated with the server-side scripting language PHP, this will create a secure system for handling a lot of event data.

The literature review will describe the current situation regarding digital event management solutions, which will also talk about advantages and disadvantages with respect to the approaches noted. The methodologies considered, and a comparison will also be made with the expected capabilities of the proposed Event Management System (EMS).

SYSTEM DESCRIPTION

The Event Management System (EMS) is a web-based application designed to streamline the process of organizing, managing, and attending events. Built with HTML, CSS, Bootstrap, JavaScript, MySQL, and PHP, the system offers a comprehensive suite of tools that cater to the needs of event organizers and attendees, providing an intuitive and responsive platform accessible from any device. The EMS architecture is designed to be modular, ensuring scalability and ease of maintenance while handling various event types, from small meetups to large conferences.

1. User Interface

Frontend Design: The interface is developed using HTML and CSS, with Bootstrap ensuring a consistent and responsive layout across different screen sizes and devices. The design focuses on a clean, user-friendly layout that makes navigation intuitive for users of all technical backgrounds.

Interactivity: JavaScript is employed to enhance the interactivity of the system, providing real-time feedback, form validation, and dynamic content loading. This improves user experience by minimizing page reloads and making interactions smooth and efficient.

2. Functional Modules

Event Creation and Management: Organizers can create and customize events by defining details such as event name, description, date, time, location, and attendee capacity. This module also allows for editing and updating event details as needed.

Registration and Ticketing: The registration module allows attendees to view upcoming events, register for events of interest, and, if applicable, purchase tickets. The system generates unique tickets with QR codes for easy check-in and tracking.

User Management: Users are categorized into different roles (organizer, attendee, admin) to control access to different functionalities. Organizers can manage attendees, send updates, and monitor registrations, while attendees can view their registered events and receive notifications.

Notifications and Reminders: Through automated email notifications, the EMS keeps attendees informed of event details, updates, and reminders. This module ensures efficient communication and timely reminders to improve attendee engagement.

Feedback Collection and Analytics: Following each event, attendees can submit feedback, which organizers can use to analyze the event's success and gather insights for improvement. The analytics dashboard provides an overview of registration numbers, attendee engagement, and feedback ratings.

3. Backend and Database

Server-Side Processing (PHP): PHP handles all server-side logic, including user authentication, session management, event processing, and interaction with the MySQL database. This enables secure and efficient data handling, ensuring that all information is processed accurately and displayed dynamically.

Database (MySQL): MySQL is used to manage data securely and reliably. The database structure is organized to store user information, event details, registration records, and feedback. Through structured queries, the system efficiently retrieves and updates information as needed.

4. Security and Access Control

User Authentication: The EMS includes a secure login system with role-based access control to differentiate between administrators, organizers, and attendees. Password encryption and session management protect user data and ensure only authorized access to sensitive functionalities. **Data Validation and Protection:** All inputs are validated on both the client and server sides to prevent SQL injection, cross-site scripting (XSS), and other security vulnerabilities. This ensures data integrity and protects user information.

5. Scalability and Flexibility

Modular Design: The EMS is built in a modular fashion, allowing for future scalability and ease of adding new features or modifying existing ones. As new requirements arise, additional modules can be integrated without disrupting the core functionality of the system.

Responsive Layout: With Bootstrap, the EMS ensures that the platform is accessible and visually consistent on devices of various screen sizes, enhancing accessibility and user experience.

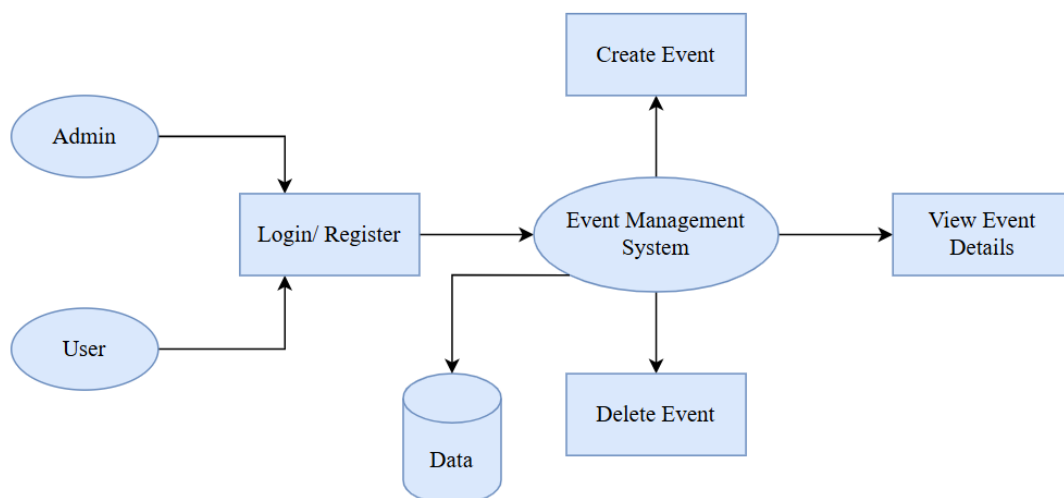
System Flow

Organizers: Log in → Create and manage events → View attendee list and send notifications → Monitor event analytics and feedback.

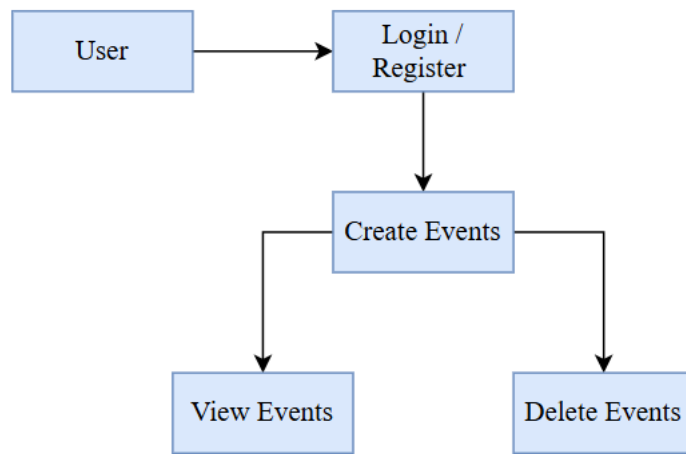
Attendees: Register or log in → View available events → Register for or purchase tickets → Receive notifications and event reminders → Provide post-event feedback.

Administrators: Log in → Oversee all events and users → Manage user permissions and data integrity.

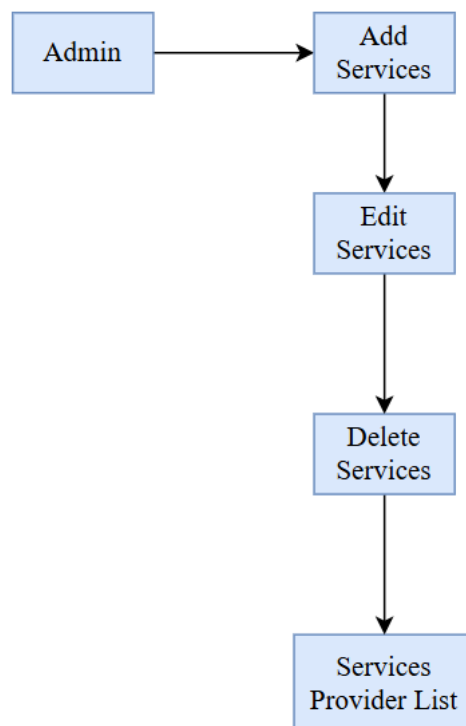
The EMS is a solution aimed at simplifying the complexities of event management, offering essential functionalities in an integrated, visually appealing, and efficient system. By automating repetitive tasks and ensuring effective communication, the EMS enhances the experience for both organizers and attendees, making it a valuable tool for modern event management.



Architectural Diagram



User Flowchart



Admin Flowchart

THEORETICAL ANALYSIS

The Event Management System (EMS) incorporates the best front-end technologies coupled with appropriate back-end technologies as well as robust database technologies; making the event management system interactive and responsive. Each tool was selected because of the strengths in developing web applications hence, enabling the development of a system that is simple to use yet functionally complex. In the following sections, we offer a brief overview of the primary technologies we employed in the EMS project and how these technologies have enabled us to construct a scalable and high-performance application.

1. HTML (HyperText Markup Language)

- HTML forms the backbone of the EMS's structure, providing the framework for displaying content on the web. As a markup language, HTML organizes content into various elements, allowing for clear, semantically structured pages. HTML is foundational to building web interfaces, and in the EMS, it is used to define the structure of forms, tables, buttons, and other essential components. Its simplicity and widespread support make it an ideal choice for building accessible and universally compatible web pages.
- **Advantages:** HTML is the foundation of web content, providing a structured framework for all page elements and defining the layout of the website. It is universally supported across all browsers, making it ideal for building accessible and platform-independent applications. Additionally, HTML's semantic tags improve SEO and accessibility, making the content more understandable to search engines and screen readers.

2. CSS (Cascading Style Sheets)

- CSS is used to style the HTML elements, controlling the look and feel of the EMS interface. CSS enables the separation of content and design, allowing for a cleaner, more maintainable codebase. In the EMS, CSS is responsible for setting visual properties like color, font, spacing, and layout, contributing to a consistent and aesthetically pleasing design. By leveraging CSS, the EMS ensures a professional, user-friendly experience that enhances accessibility and usability across devices.
- **Advantages:** CSS allows for efficient separation of content and design, enhancing code organization and reusability. It enables developers to define style rules for different elements, which can be easily modified to change the look and feel of the entire site. CSS provides control over the visual aspects of the page, such as colors, fonts, and layouts, making it essential for creating a cohesive and visually appealing interface.

3. Bootstrap

- Bootstrap is a popular front-end framework that simplifies the development of responsive and mobile-first web applications. It provides pre-built CSS and JavaScript components, such as grids, buttons, and navigation bars, which streamline the design process and ensure a consistent layout. In the EMS, Bootstrap is used to create a responsive interface that adjusts dynamically to different screen sizes and devices, providing users with a seamless experience on desktops, tablets, and smartphones. The use of Bootstrap reduces development time and enhances the project's scalability by standardizing design components.
- **Advantages:** Bootstrap is a powerful front-end framework that enables developers to create responsive, mobile-first designs with ease. Its pre-designed components (like buttons, modals, and navigation bars) save development time, while its grid system makes it easy to create layouts that adjust seamlessly to different screen sizes. Bootstrap also ensures a uniform design across the application, making it ideal for creating aesthetically pleasing, responsive interfaces.

4. JavaScript

- JavaScript is a versatile programming language that allows for dynamic and interactive web experiences. It is essential for client-side scripting in the EMS, enabling features such as form validation, real-time data updates, and interactive elements. JavaScript provides asynchronous functionality, allowing parts of the page to update without requiring a full reload. In the EMS, JavaScript enhances usability by making the system more responsive and interactive, delivering a smooth, user-friendly experience that mimics native applications.
- **Advantages:** JavaScript is a versatile language that allows for client-side interactivity, making web pages dynamic and engaging. It can run in the user's browser, providing a fast, responsive experience without needing to reload the page. JavaScript is also highly adaptable, compatible with various libraries and frameworks, and can easily interact with HTML and CSS, making it indispensable for modern web applications.

5. PHP (Hypertext Preprocessor)

- PHP is a widely-used server-side scripting language that powers the backend of the EMS. PHP excels in web development due to its compatibility with HTML and databases, ease of use, and extensive community support. In the EMS, PHP handles essential server-side functions, including user authentication, session management, data processing, and interactions with the MySQL database. PHP allows for secure data handling and dynamic

content generation, making it a key component in creating a robust, interactive, and efficient system.

- **Advantages:** PHP is a widely-used server-side scripting language known for its ease of use and strong integration with HTML. It is well-suited for web applications that require dynamic content generation and secure data processing. PHP is highly compatible with MySQL, making it a preferred choice for applications that involve frequent database interactions. It also supports various frameworks and libraries, further enhancing its versatility.

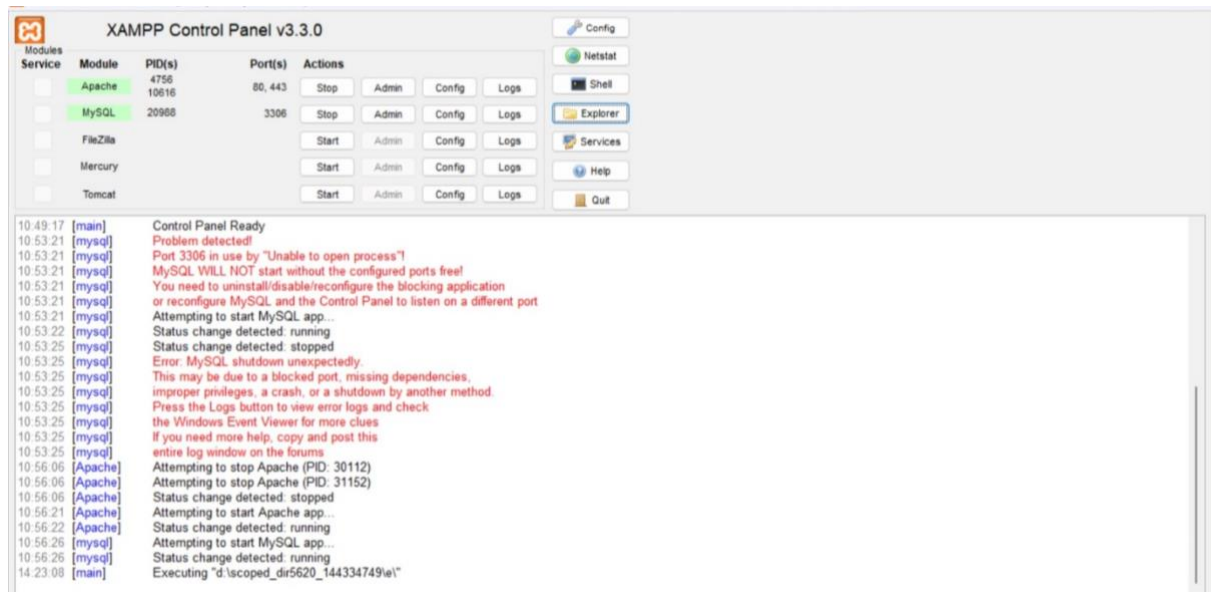
6. MySQL

- MySQL is a relational database management system (RDBMS) used to store and manage data for the EMS. MySQL is known for its performance, reliability, and scalability, making it suitable for handling large volumes of data and supporting concurrent users. The EMS relies on MySQL to store event details, user information, registration data, and feedback securely and efficiently. Structured Query Language (SQL) queries facilitate data retrieval and manipulation, allowing the EMS to update, delete, and display data as needed in real time. The use of MySQL ensures that data is accessible and well-organized, contributing to the stability and performance of the EMS.
- **Advantages:** MySQL is a powerful and reliable relational database management system that offers scalability, speed, and secure data handling. Known for its performance and compatibility with a wide range of programming languages (including PHP), MySQL can handle large volumes of data and concurrent users. It is also open-source, making it a cost-effective solution for projects of various scales.

7. Apache HTTP Server

- **Advantages:** Apache HTTP Server (often referred to simply as Apache) is one of the most widely used and reliable web servers available today. Known for its stability, flexibility, and security, Apache can handle large amounts of traffic while providing robust performance. It supports various modules, allowing customization for different applications and enabling features such as URL redirection, SSL encryption, and load balancing. Additionally, Apache is open-source, highly compatible with PHP, and has extensive community support, making it a preferred choice for web development projects.
- **Role in EMS:** In the EMS, Apache serves as the web server that hosts the application, handling requests from clients and delivering web pages efficiently. It manages communication between the client-side (HTML, CSS, JavaScript) and the server-side (PHP, MySQL) components, ensuring smooth data flow and secure access. By serving PHP files and managing session handling, Apache enables the EMS to provide a reliable and high-performance environment, allowing organizers and attendees to interact with the

system seamlessly and securely. Apache's modular nature also allows for future scalability, giving the system room to grow and handle increased traffic as more users join the platform.



Apache Server Monitor

8. System Integration and Data Flow

- Advantages:** By combining Apache with HTML, CSS, JavaScript, PHP, Bootstrap, and MySQL, the EMS benefits from a robust and efficient architecture. Apache acts as the intermediary, ensuring that client requests are processed smoothly and securely by PHP, which interacts with MySQL to retrieve or store data. Together, these tools create a responsive, interactive, and scalable platform capable of meeting the demands of modern event management applications.
- Role in EMS:** Apache acts as the backbone of the EMS server environment, receiving and responding to client requests while managing data flow between the client and server sides. HTML, CSS, and Bootstrap provide a responsive interface; JavaScript ensures interactivity; PHP handles server-side logic; and MySQL securely stores data. Apache integrates all these components, supporting the EMS as a unified, reliable platform for event organization and attendee interaction. This integration allows for high availability, performance, and a seamless experience for all users, creating an efficient solution for event management.

METHODOLOGY

1. System Design

- **Process:**
 - **Frontend Design:** Use HTML and CSS for structuring content and styling, with Bootstrap for responsive design. Design wireframes to outline key UI components like the event dashboard, registration forms, and feedback pages.
 - **Backend Architecture:** Define server-side processes using PHP, focusing on modularity to enable code reusability and maintainability. Design the MySQL database schema to store event details, user information, registration records, and feedback data.
 - **Web Server Setup:** Configure the Apache HTTP server to host the application, including necessary modules for handling PHP requests and SSL encryption for secure communication.
- **Output:** A complete system architecture design, including UI wireframes, database schema, and server setup guidelines, providing a clear development roadmap.

2. Frontend Development

- **Process:**
 - Use **HTML** to structure the pages, creating forms, tables, and navigation components.
 - Apply **CSS** and **Bootstrap** to style and enhance the layout, ensuring responsiveness across devices.
 - Incorporate **JavaScript** for interactive features, such as form validation, dynamic content updates, and real-time feedback.
- **Output:** A visually appealing, responsive frontend that provides easy navigation, clear information display, and smooth interactions.

3. Backend Development

- **Process:**
 - Develop PHP scripts to handle core operations, such as user authentication, event creation, registration management, and ticket generation.
 - Use PHP to communicate with the MySQL database, executing SQL queries to insert, update, delete, or retrieve data as needed.

- Set up session management and role-based access control to secure user data and limit access to specific functionalities based on user roles (admin, organizer, attendee).
- **Output:** A reliable backend system that handles data operations and business logic, enabling dynamic and secure user interactions with the EMS.

4. Database Development

- **Process:**
 - Create tables for each entity (e.g., events, users, registrations, feedback) with appropriate fields and relationships.
 - Implement data constraints and indexing to ensure data integrity and optimize query performance.
 - Configure backup mechanisms for data recovery and security measures to protect sensitive user information.
- **Output:** A well-structured MySQL database that enables fast, reliable data retrieval and storage, supporting efficient application performance.

5. Server Configuration and Deployment

- **Process:**
 - Install and configure Apache with necessary modules to support PHP processing and SSL encryption.
 - Optimize server settings to handle concurrent users and efficiently process client requests.
 - Deploy the EMS to the server, setting up domain configurations, enabling secure (HTTPS) connections, and testing server performance.
- **Output:** A fully configured and deployed server environment that reliably hosts the EMS, providing secure and efficient access to all users.

RESULT AND DISCUSSION

Database code

```
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
```

```
SET time_zone = "+00:00";
```

```
CREATE TABLE `events` (  
  `EventID` int(11) NOT NULL,  
  `Title` varchar(255) NOT NULL,  
  `Description` varchar(255) NOT NULL,  
  `StartDate` varchar(255) NOT NULL,  
  `EndDate` varchar(255) NOT NULL,  
  `Cost` int(11) NOT NULL,  
  `LocationID` int(11) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
INSERT INTO `events` (`EventID`, `Title`, `Description`, `StartDate`, `EndDate`, `Cost`,  
  `LocationID`) VALUES
```

```
(1, 'Wedding Anniversary', '1st Anniversary Celebration', '10-Oct-2015', '10-Oct-2016',  
  25000, 1);
```

```
CREATE TABLE `locations` (  
  `LocationID` int(11) NOT NULL,  
  `Name` varchar(255) NOT NULL,  
  `Address` varchar(255) NOT NULL,  
  `ManagerFName` varchar(255) NOT NULL,  
  `ManagerLName` varchar(255) NOT NULL,
```



```
`ManagerEmail` varchar(255) NOT NULL,  
`ManagerNumber` int(11) NOT NULL,  
`MaxCapacity` int(11) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
INSERT INTO `locations` (`LocationID`, `Name`, `Address`, `ManagerFName`,  
`ManagerLName`, `ManagerEmail`, `ManagerNumber`, `MaxCapacity`) VALUES  
(1, 'Royal Hotel', 'Bray', 'John', 'Byrne', 'John@email.com', 123456, 100);
```

```
CREATE TABLE `users` (  
`username` varchar(255) NOT NULL,  
`password` varchar(255) NOT NULL,  
`role` varchar(255) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
INSERT INTO `users` (`username`, `password`, `role`) VALUES  
('test', '1234', 'user');
```

```
ALTER TABLE `events`  
ADD PRIMARY KEY (`EventID`),  
ADD KEY `LocationID` (`LocationID`);
```

```
ALTER TABLE `locations`  
ADD PRIMARY KEY (`LocationID`);
```

```
ALTER TABLE `events`
```

```
    MODIFY `EventID` int(11) NOT NULL AUTO_INCREMENT,  
    AUTO_INCREMENT=2;
```

```
ALTER TABLE `locations`
```

```
    MODIFY `LocationID` int(11) NOT NULL AUTO_INCREMENT,  
    AUTO_INCREMENT=2;
```

“Event” class code

```
<?php
```

```
class Event {
```

```
    private $title;
```

```
    private $description;
```

```
    private $startDate;
```

```
    private $endDate;
```

```
    private $cost;
```

```
    private $locationID;
```

```
    public function __construct($id, $title, $description, $sDate, $eDate, $cost, $locID) {
```

```
        $this->id = $id;
```

```
        $this->title = $title;
```

```
        $this->description = $description;
```

```
        $this->startDate = $sDate;
```

```
        $this->endDate = $eDate;
```

```
        $this->cost = $cost;
```

```
        $this->locationID = $locID;
```

```
    }
```

```
    public function getId() { return $this->id; }
```

```
    public function getTitle() { return $this->title; }
```

```
    public function getDescription() { return $this->description; }
```

```

    public function getStartDate() { return $this->startDate; }
    public function getEndDate() { return $this->endDate; }
    public function getCost() { return $this->cost; }
    public function getLocationID() { return $this->locationID; }
}
?>

```

“CreateEvent ” code

```

<?php
require_once 'classes/Event.php';
require_once 'classes/EventTableGateway.php';
require_once 'classes/Connection.php';
require_once 'validateEvents.php';

$formdata = array();
$errors = array();

validateEvents(INPUT_POST, $formdata, $errors);

if (empty($errors)) {
    $title = $formdata['Title'];
    $description = $formdata['Description'];
    $startDate = $formdata['StartDate'];
    $endDate = $formdata['EndDate'];
    $cost = $formdata['Cost'];
    $locID = $formdata['LocID'];

    $event = new Event(-1, $title, $description, $startDate, $endDate, $cost, $locID);

    $connection = Connection::getInstance();

    $gateway = new EventTableGateway($connection);

```

```

        $id = $gateway->insert($event);

        header('Location: viewEvents.php');
    }
    else {
        require 'createForm.php';
    }

```

“viewEvent” code

```

<?php
require_once 'classes/Event.php';
require_once 'classes/EventTableGateway.php';
require_once 'classes/Connection.php';

if (!isset($_GET['id'])) {
    die("Illegal request");
}

$id = $_GET['id'];

$connection = Connection::getInstance();
$gateway = new EventTableGateway($connection);

$stmt = $gateway->getEventsById($id);

$row = $stmt->fetch(PDO::FETCH_ASSOC);
if (!$row) {
    die("Illegal request");
}

?>
<!DOCTYPE html>

```

```

<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title></title>
    <?php require 'utils/styles.php'; ?>
    <?php require 'utils/scripts.php'; ?>
  </head>
  <body>
    <?php require 'utils/header.php'; ?>
    <div class = "content">
      <div class = "container">
        <?php
          if (isset($message)) {
            echo '<p>'.$message.'</p>';
          }
        ?>
        <table class = "table table-hover">
          <thead><!--table labels-->
            <tr>
              <th>Event ID</th>
              <th>Title</th>
              <th>Description</th>
              <th>Start Date</th>
              <th>End Date</th>
              <th>Cost</th>
              <th>Location ID</th>
              <th>Actions</th>
            </tr>
          </thead>
          <tbody><!--table contents, pulled from database-->
            <?php
              echo '<tr>';
              echo '<td>' . $row['eventID'] . '</td>';

```

```

        echo '<td>' . $row['Title'] . '</td>';
        echo '<td>' . $row['Description'] . '</td>';
        echo '<td>' . $row['StartDate'] . '</td>';
        echo '<td>' . $row['EndDate'] . '</td>';
        echo '<td>' . $row['Cost'] . '</td>';
        echo '<td>' . $row['locationID'] . '</td>';
        echo '<td>'
            . '<a class="delete"
href="deleteEvent.php?id='.$row['eventID'].'">Delete</a> '
            . '</td>';
        echo '</tr>';
    }
}
?>
</tbody>
</table>

<a class="btn btn-default" href="viewEvents.php"><span class="glyphicon
glyphicon-circle-arrow-left"></span> Back</a>

</div>
</div>
<?php require 'utils/footer.php'; ?>
</body>
</html>

```

“Login” code

```

<?php

require_once 'utils/functions.php';
require_once 'classes/User.php';
require_once 'classes/DB.php';
require_once 'classes/UserTable.php';

start_session();

try {
    $formdata = array();

```

```
$errors = array();
```

```
$input_method = INPUT_POST;
```

```
$formdata['username'] = filter_input($input_method, "username",  
FILTER_SANITIZE_STRING);
```

```
$formdata['password'] = filter_input($input_method, "password",  
FILTER_SANITIZE_STRING);
```

```
if (empty($formdata['username'])) {
```

```
    $errors['username'] = "Username required";
```

```
}
```

```
$email = filter_var($formdata['username'], FILTER_VALIDATE_EMAIL);
```

```
if ($email != $formdata['username']) {
```

```
    $errors['username'] = "Valid email required";
```

```
}
```

```
if (empty($formdata['password'])) {
```

```
    $errors['password'] = "Password required";
```

```
}
```

```
if (empty($errors)) {
```

```
    $username = $formdata['username'];
```

```
    $password = $formdata['password'];
```

```
$connection = DB::getConnection();
```

```
$userTable = new UserTable($connection);
```

```
$user = $userTable->getUserByUsername($username);
```

```
if ($user == null) {
```

```
    $errors['username'] = "Username is not registered";
```

```
}
```

```
else {
```

```
    if ($password !== $user->getPassword()) {
```

```
        $errors['password'] = "Password is incorrect";
```

```

    }
}

if (!empty($errors)) {
    throw new Exception("");
}

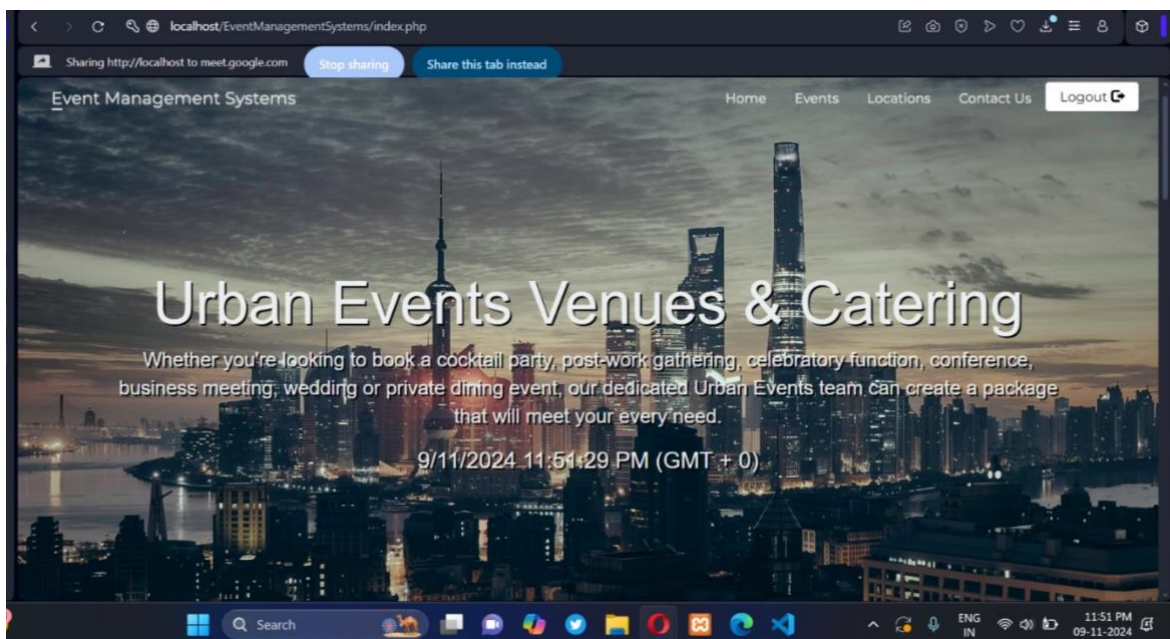
$_SESSION['user'] = $user;

header('Location: index.php');
}

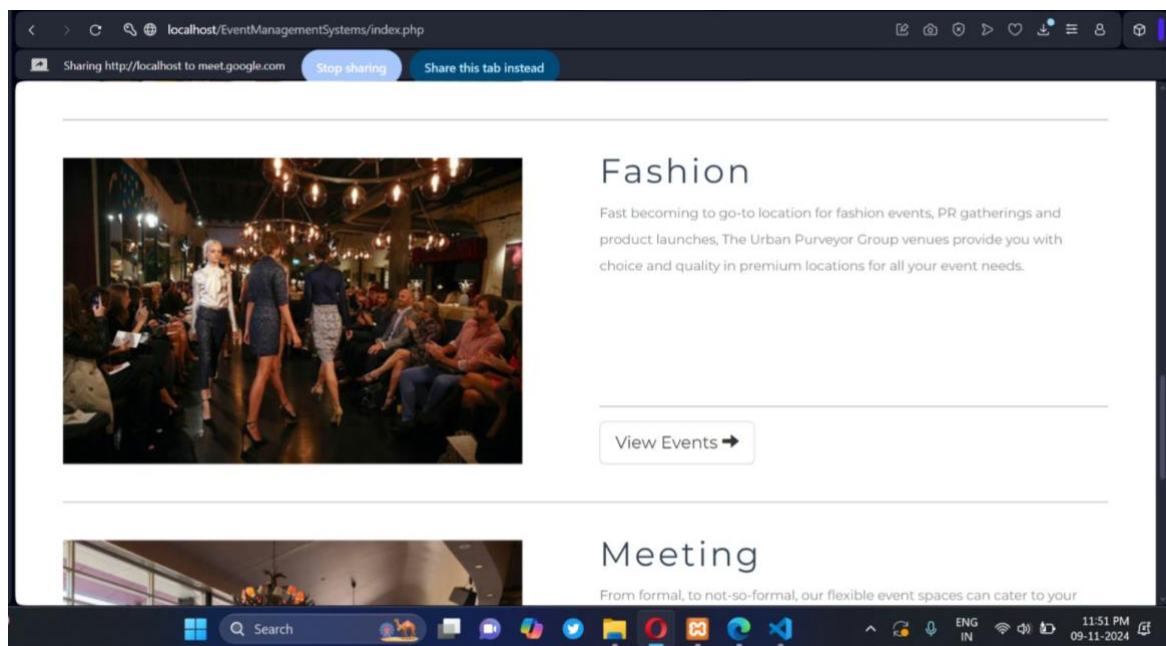
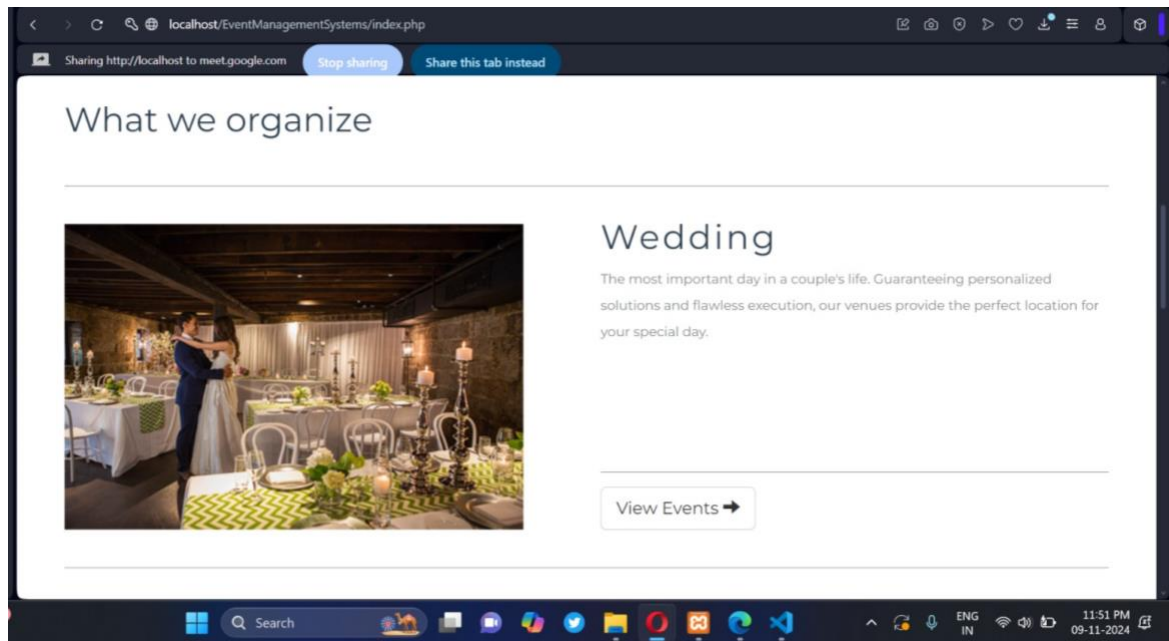
catch (Exception $ex) {
    $errorMessage = $ex->getMessage();
    require 'login_form.php';
}

?>

```



Home page



Home pages

Location ID	Name	Address	Manager First Name	Manager Last Name	Manager Email	Manager Number	Max Capacity	Actions
1	Royal Hotel	Bray	John	Byrne	John@email.com	123456	100	View Edit Delete
2	kolkata	sectorV	Shirsendu	Roy	shirsenduroysr@gmail.com	2147483647	100	View Edit Delete

Create Location

View Events page

CONCLUSION

The Event Management System (EMS) successfully provides an efficient, user-friendly solution for managing events in a streamlined, digital format. Using a combination of front-end and back-end technologies HTML, CSS, Bootstrap, JavaScript, PHP, MySQL, and Apache the system allows for seamless communication between organizers and attendees, ensuring smooth event creation, registration, ticketing, and feedback processes.

This project follows a structured methodology, beginning with requirements gathering and continuing through to final deployment and maintenance. By focusing on scalability and security, the EMS is designed to handle the needs of both small and large events, offering a reliable platform for users. Apache as the server backbone and MySQL for database management have been instrumental in achieving an environment that supports multiple users while maintaining speed and stability. The application's modular design also facilitates future updates, allowing it to adapt to new requirements as they arise.

Ultimately, the EMS project highlights the effectiveness of modern web technologies in solving real-world problems, particularly in the field of event management. By reducing the complexity and workload associated with organizing events, the EMS provides a valuable tool for organizers and enhances the experience for attendees. This foundation not only meets current needs but also leaves room for further development, making it a forward-looking solution in the digital landscape of event management.