Event Management Website Project Report

**By**

**Manashri Shevade -AF04955268**

**Mayida ulde-AF04957232**

**Index**

|  |  |  |
| --- | --- | --- |
| **Sr.no** | **Topic** | **Page no** |
| 1 | Title of Project | 1 |
| 2 | Acknowledgement | 3 |
| 3 | Abstract | 4 |
| 4 | Introduction | 5 |
| 5 | Objective | 6 |
| 6 | System Analysis | 7 |
| 7 | System Design | 22 |
| 8 | Screenshots | 32 |
| 9 | Coding | 39 |
| 10 | Testing | 46 |
| 11 | Report | 47 |
| 12 | Future Scope | 48 |
| 13 | Conclusion | 48 |
| 14 | Bibliography | 49 |
| 15 | References | 49 |

# Acknowledgement

The project **“Event Management Website”** is the Project work carried out by

|  |  |
| --- | --- |
| **Name** | **Enrollment No** |
| **Manashri Shevade** | **AF04955268** |
| **Mayida ulde** | **AF04957232** |

Under the Guidance.

We are thankful to my project guide for guiding me to complete the Project.

His suggestions and valuable information regarding the formation of the Project Report have provided me a lot of help in completing the Project and its related topics.

We are also thankful to my family member and friends who were always there to provide support and moral boost up.

# Abstract

The project called **Event Management Website** is about creating a web application to streamline the process of organizing and managing events. This system replaces traditional manual methods with a digital platform where administrators can easily add, edit, delete, and manage event details, registrations, and attendees. It also stores event data for a long time, making it easier and faster to handle.

This portal lets users browse events by category, read detailed event descriptions, and register for events. It also includes features for secure user authentication and event tracking. This helps administrators understand event participation better and manage logistics efficiently.

The system is simple, easy to use, and works well on various devices. It allows real-time updates, registration management, and better analysis of event data. It helps organizations save time and resources while reaching more people. This platform focuses on efficiency, accessibility, and making decisions based on event participation feedback.

# Introduction

In today’s age of Information Communication and Technology, we are surrounded by technology at every moment. From the moment we wake up to the time we go to bed, technology plays a crucial role in our lives. One of the most important aspects of this technological revolution is the development of web applications. These applications enable us to access information from anywhere, anytime, and at a low cost. Information has become one of the most valuable resources in the modern world, and web applications, such as event management portals, provide us with up-to-date information and tools for organization.

Our project aims to develop an Event Management Website using React.js that makes the latest event information accessible to people at any time. The portal will allow users to stay informed about upcoming events and other important updates in a timely and dynamic manner. Additionally, robust event management functionality is integrated into the portal, enabling administrators to efficiently create, manage, and track events, registrations, and attendees. This adds depth to event organization and enhances administrative insights for logistics and planning.

## Objective of the Present Work :

The objectives of this project are as follows:

* To develop a web application for an Event Management Website that can simplify event organization.
* To provide a platform for users to browse, search, and register for events.
* To enable administrators to efficiently manage event details, categories, and user registrations.
* To deliver an intuitive, visually appealing, and engaging user interface using React.js.
* To ensure secure user authentication and data management.
* To create a dynamic platform that allows for easy addition and modification of event information without any complexity.

# System analysis

* 1. **PROBLEM DEFINITION**

Many event organizers do not have an easy way to manage event content, registrations, and attendee information. Users often find it difficult to discover events and register seamlessly. Also, traditional methods of organizing events take a lot of time and effort, leading to inefficiencies and potential errors.

This project solves these problems by allowing admins to manage events easily and providing a user-friendly platform for event discovery and registration. The system makes event organization faster, helps admins organize content better, and gives users a platform to find and participate in events efficiently.

* 1. **Preliminary Investigation Purpose**

The Event Management Website is designed to make event organization and discovery accessible anytime and anywhere through a digital platform. It aims to replace traditional event management methods by allowing administrators to manage event details and registrations efficiently while providing users with real-time updates and interactive features like event browsing and registration.

**Benefits**

The portal provides several advantages:

1. Instant Event Access: Users can browse event articles categorized by topics such as conferences, workshops, entertainment, and sports.
2. Efficient Content Management: Admins can easily add, edit, delete, and restore event details without delays.
3. User Engagement: Readers can interact by registering for events and receiving updates.
4. Improved Management: The system helps admins monitor user registrations and manage event logistics effectively.

**Proposed System**

The proposed system offers a structured, reliable, and automated approach to event management:

* + - Users can browse events, register for events, and receive instant updates.
    - Administrators can manage events, categories, and analyze registration data.
    - The portal is designed to be simple, responsive, and efficient, ensuring a smooth experience for both users and administrators.
  1. **Feasibility Study**

The feasibility study evaluates whether the Event Management Website using React.js project is practical, achievable, and beneficial. This assessment ensures the system can be successfully implemented within available resources, technology, and constraints.

**Types of Feasibility Analysis**

### Technical Feasibility

The portal is built using React.js for frontend and [Your chosen backend technology, e.g., Node.js (Express)] for backend, making it compatible with existing web technologies.

The database is managed using [Your chosen database, e.g., MongoDB], ensuring secure and scalable data storage.The system is hosted on a reliable web server, supporting high user traffic.Economic Feasibility Open-source technologies like React.js and help minimize software costs. The automated event management features save administrative effort in organizing and tracking events.

### Operational Feasibility

1. Users can easily access the portal through any device, ensuring high usability.
2. Events are categorized, making navigation simple.
3. Admins can efficiently manage events and registrations, ensuring a smooth process for organizers.

**Schedule Feasibility**

1. The development timeline is realistic, covering design, coding, testing, and deployment within estimated deadlines.
2. Modular development ensures different parts of the system are built incrementally, improving flexibility.

### Schedule Feasibility

1. The development timeline is realistic, covering design, coding, testing, and deployment within estimated deadlines.
2. Modular development ensures different parts of the system are built incrementally, improving flexibility.

### Social Feasibility

1. The project promotes organized event participation among users.
   * + It fosters engagement without logistical hurdles.
     + The system is accessible to all users, regardless of technical expertise.
   1. Project Planning

Purpose of Project Planning

Project planning ensures that the development of the Event Management Website using React.js follows a structured approach. It helps define the workflow, resource allocation, timelines, and risks involved to deliver the system efficiently.

Phases Covered in the Plan

The planning is divided into different phases to ensure smooth execution:

1. Preliminary Investigation – Understanding the project scope and objectives.

2. System Analysis– Identifying challenges, gathering requirements, and defining solutions.

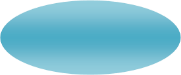
3. System Design – Structuring modules, database design, and UI development.

4. Coding – Developing the portal using React.js for frontend and [Your chosen backend] for backend.

5. Security – Implementing authentication, data encryption, and user privacy measures.

6. Testing– Performing unit testing, integration testing, and user acceptance testing.

7. Implementation – Deploying the final system and ensuring smooth operation.



**Start**

Preliminary

System Analysis

System Design

Coding

Testing

Security

Implementation

**Stop**

* 1. Software Requirement Specification (SRS)

The Software Requirement Specification (SRS) outlines the fundamental requirements of the Event Management Website using React.js to ensure efficient functionality, usability, and maintainability.

System Overview

The Event Management Website allows users to access and interact with event details across multiple categories. The system is structured into three modules:

1. User Module – Enables users to browse events, search for events, and register for them.

2. Admin Module– Provides tools for event management, user management, and registration tracking.

3. Event Module – Manages the creation, modification, and publication of event details.

Stop

Start

Preliminary

System Analysis

System Design

Coding

Testing

Security

Implementation

Software & Hardware Requirements Software Requirements

Frontend: React.js, HTML, CSS, JavaScript for responsive UI

Backend: Node.js (Express)for data handling and security

Database : MySQL for storing event details, registrations, and user details

Web Server: Apache or Nginx for hosting (or specific server for backend, e.g., Node.js server)

Development Environment: Node.js, npm/yarn, code editor (e.g., VS Code)

Hardware Requirements

1. Processor: Intel i5 or higher
2. RAM: Minimum 8GB
3. Storage: At least 100GB for database and media files
4. Connectivity: Internet access for real-time updates
   1. Functional Requirements

1. User Module

Users can:

* Read event details categorized into Conferences, Workshops, Entertainment, Sports, etc.
* Search for specific events using keywords.
* Register for events.
* View their registered events.
* Login and signup for an account.

2. Admin Module

Admins have full control over the platform and can:

* Secure Login System – Access the admin dashboard through an authentication system.
* Dashboard Management – Control event categories, and published events.
* User Management – View and manage registered users.
* Event Management – Add, edit, delete, and restore event posts.
* Registration Management – View and manage event registrations.
* Page Management – Manage static pages like 'About Us' and 'Contact Us'.

12 3. Event Module

This module handles the core event functionalities:

* Create new events with details like title, description, date, time, location, capacity, and image.
* Update existing event details.
* Delete events.
* Publish or unpublish events.
* Manage event categories.
  1. ​Software Engineering Paradigm

The development of the Event Management Website using React.js follows a structured approach to ensure efficiency, reliability, and maintainability. The chosen paradigm helps streamline the project by defining clear phases while allowing iterative improvements

Development Model: Adapted Waterfall Model

The Waterfall Model is traditionally a linear approach, but for this project, an iterative feedback mechanism is incorporated. This helps refine earlier phases based on insights gathered during implementation.

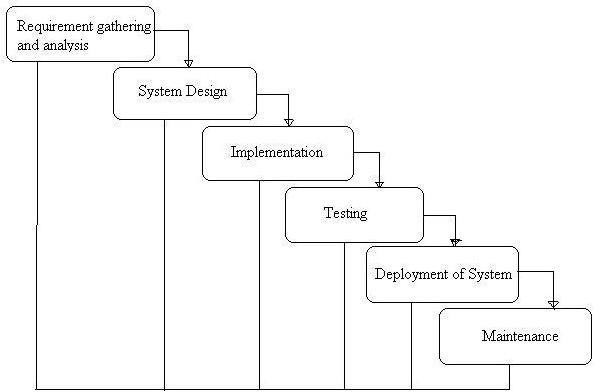
Key Adaptations in the Waterfall Model:

1. Structured Phase Progression – Each stage follows a defined sequence, ensuring clarity in execution.

2. Iterative Refinements – Feedback loops allow adjustments, especially between testing and coding.

3. Defined Milestones – Each stage reaches completion before moving to the next phase.

4. Flexible Adjustments – Overlapping is permitted when necessary to enhance efficiency.



Phases of Development

1. Requirement Analysis & System Study

* + - Identifying project goals, challenges, and functional specifications.
    - Gathering stakeholder requirements and defining core functionalities.

2. System Design

* + - Structuring the database, modules, and architecture.
    - Designing user interfaces for optimal accessibility.

3. Implementation (Coding)

* + - Frontend development using React.js, HTML, CSS, Bootstrap.
    - Backend development using Node.js.
    - Database integration with MySQL

4. Testing & Debugging

* + - Unit testing, integration testing, and usability checks.
    - Debugging for performance improvements.

5. Deployment & Maintenance

* + - Hosting on a scalable environment.
    - Continuous updates for feature enhancements.
  1. Data Flow Diagram:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both. It shows how data enter and leaves the system, what changes the information, and where data is stored.

The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called as a data flow graph or bubble chart.

**The following observations about DFDs are essential:**

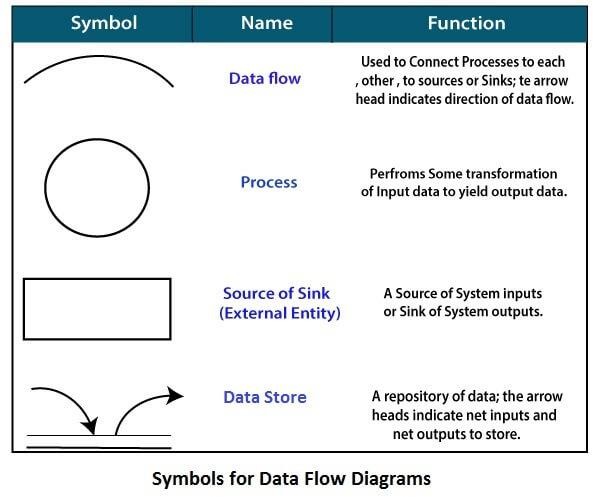
1. All names should be unique. This makes it easier to refer to elements in the DFD.

2. Remember that DFD is not a flow chart. Arrows in a flow chart represent the order of events; arrows in DFD represent flowing data. A DFD does not involve any order of events.

3. Suppress logical decisions. If we ever have the urge to draw a diamond-shaped box in a DFD, suppress that urge! A diamond-shaped box is used in flow charts to represent decision points with multiple exists paths of which only one is taken. This implies an ordering of events, which makes no sense in a DFD.

4. Do not become bogged down with details. Defer error conditions and error handling until the end of the analysis.

Standard symbols for DFDs are derived from the electric circuit diagram analysis and are shown in fig:

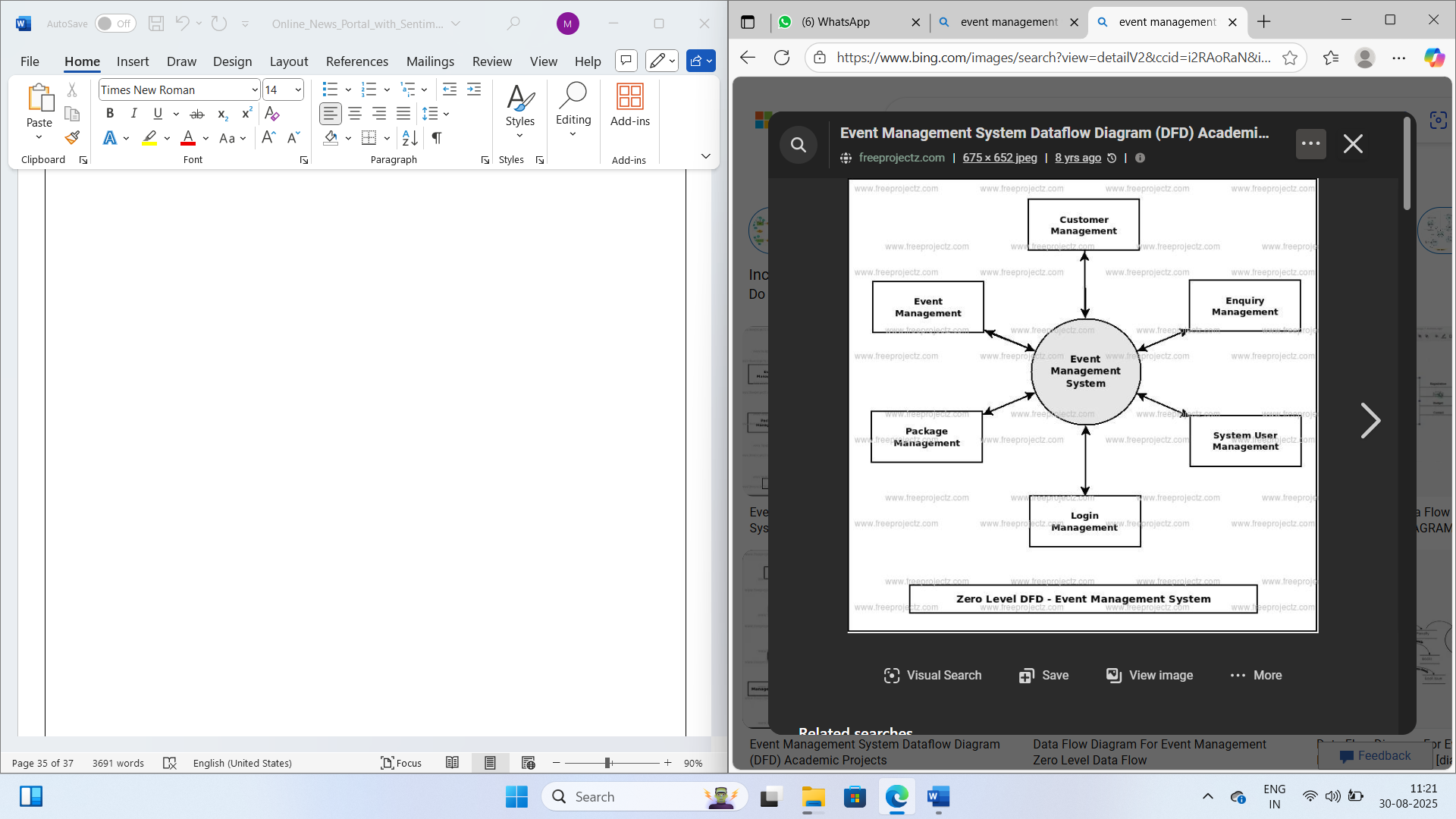


Circle: A circle (bubble) shows a process that transforms data inputs into data outputs.

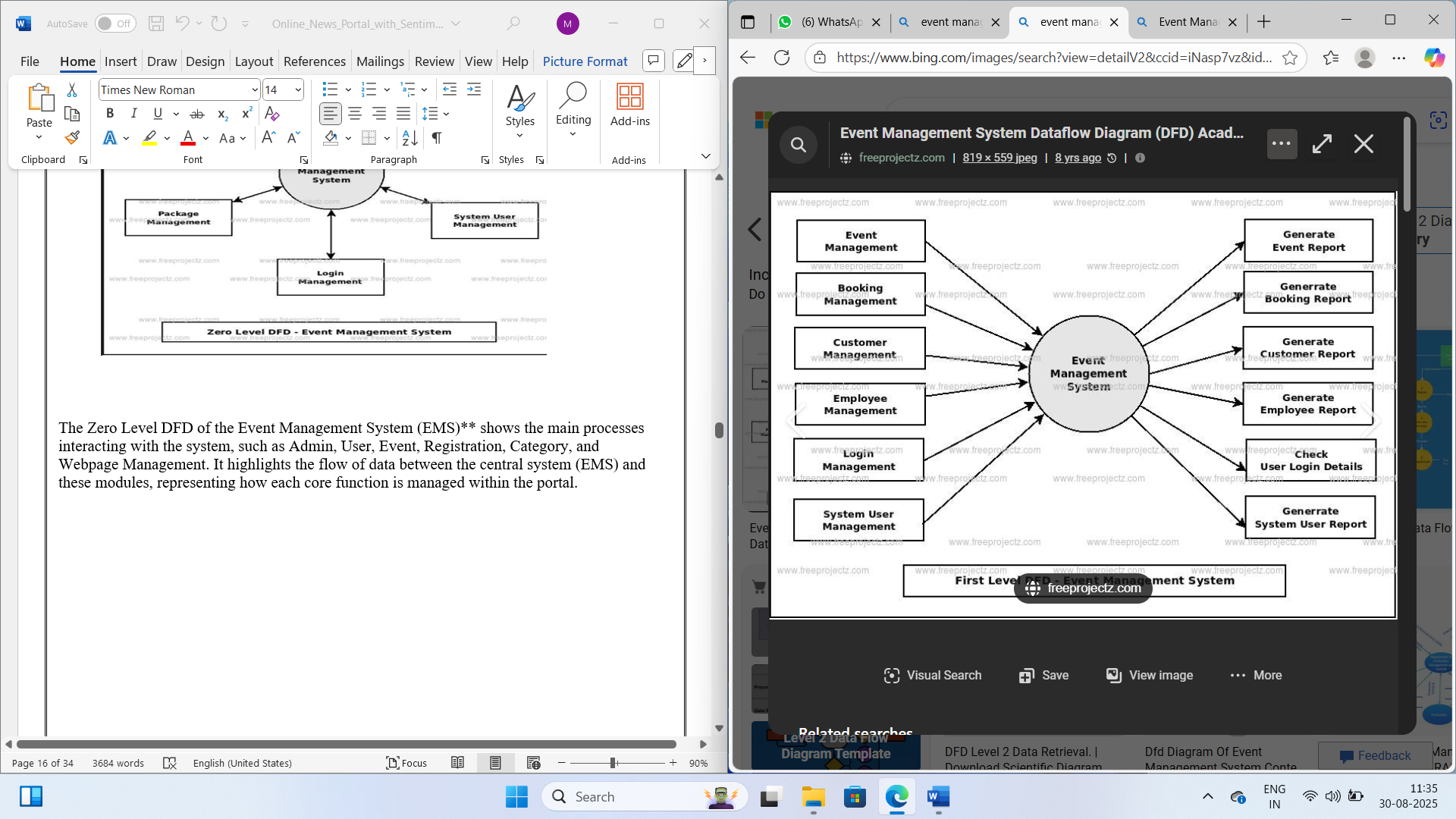
Data Flow: A curved line shows the flow of data into or out of a process or data store.

Data Store: A set of parallel lines shows a place for the collection of data items. A data store indicates that the data is stored which can be used at a later stage or by the other processes in a different order. The data store can have an element or group of elements.

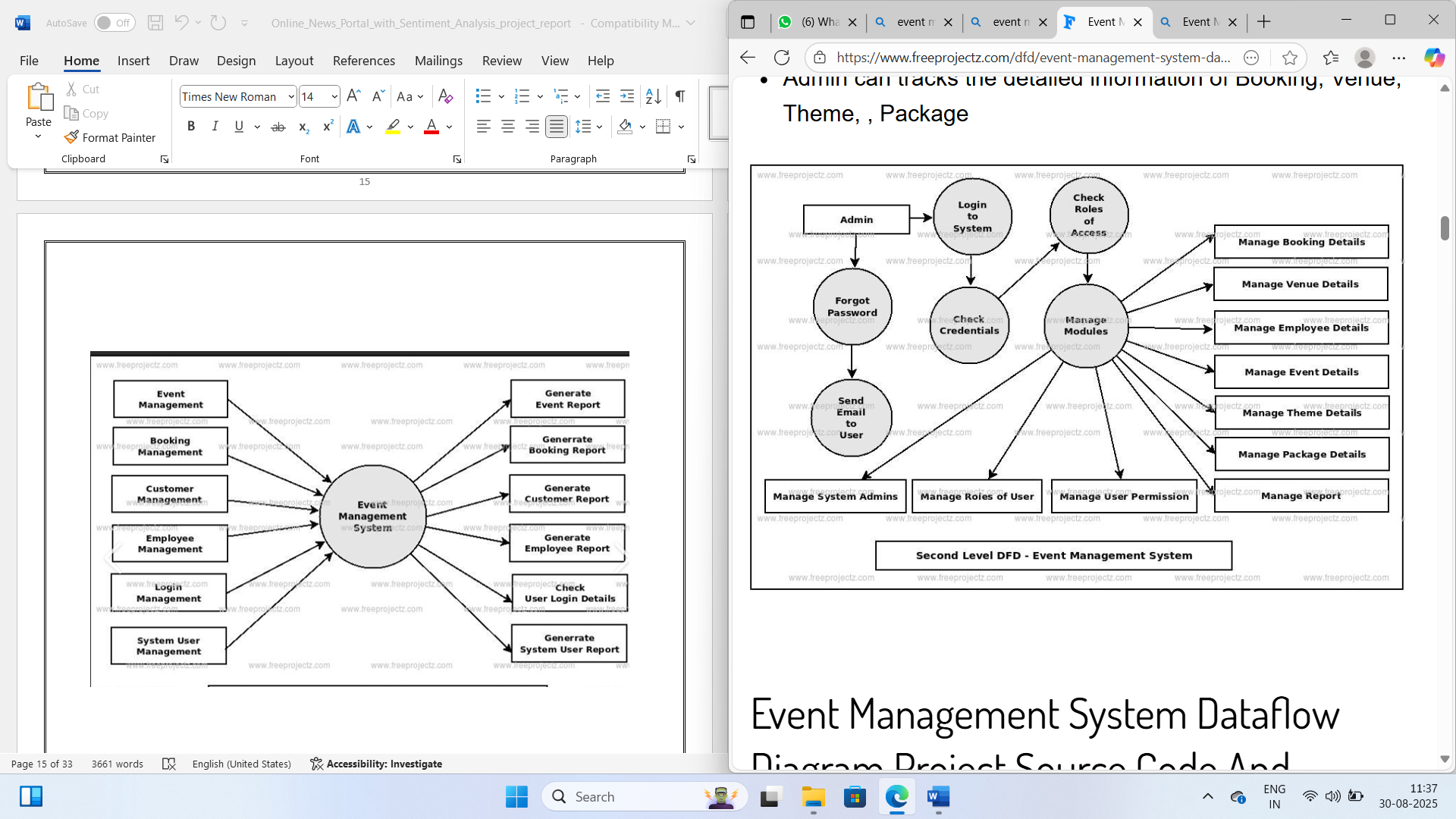
Source or Sink: Source or Sink is an external entity and acts as a source of system inputs or sink of system outputs.



The Zero Level DFD of the Event Management System (EMS)\*\* shows the main processes interacting with the system, such as Admin, User, Event, Registration, Category, and Webpage Management. It highlights the flow of data between the central system (EMS) and these modules, representing how each core function is managed within the portal.



The First-Level DFD of EMS shows how the system handles key functions like user authentication, admin management, and connects them to content-related modules such as events, categories, and user registrations for smooth portal operations.

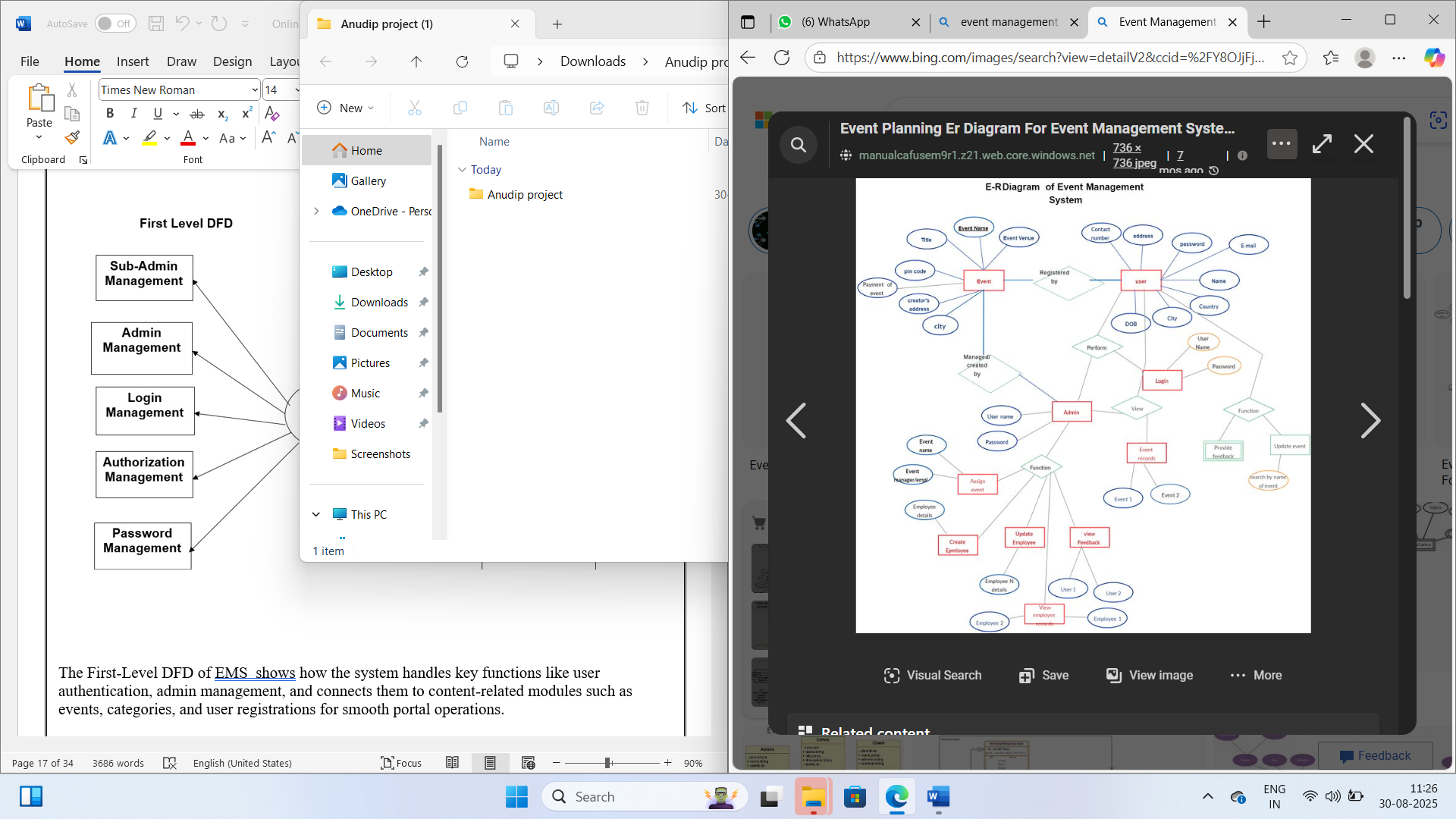


The Second-Level DFD shows the internal working of the admin in the EMS system. It includes processes like login, credential checking, role-based access, and module management. The admin can manage users, categories, events, and registrations, along with profile updates and password changes.

This Second-Level DFD represents the internal workflow for the User in the EMS system. After logging in (or as a guest), the system allows them to browse events, view details, and register for events. Additionally, the User can manage their profile and view their registered events

ER diagram

The ER Diagram of the Event Management System shows key relationships between Users, Admins, Events, Categories, and Registrations, ensuring structured data management and insightful moderation.



1. System design
   1. Module

Users can:

\* Read event details categorized into Conferences, Workshops, Technology, Sports, Entertainment, etc.

\* Search for specific events using keywords.

\* Register for events.

\* View their registered events.

1. Admin Module

Admins have full control over the platform and can:

\*Secure Login System – Access the admin dashboard through an authentication system.

\*Dashboard Management– Control event categories, and published events.

\*User Management – View and manage registered users.

\*Event Management – Add, edit, delete, and restore event posts.

\*Registration Management – View and manage event registrations.

\*Page Management – Manage static pages like 'About Us' and 'Contact Us'.

3. Event Module

This module handles the core event functionalities:

\* Create new events with details like title, description, date, time, location, capacity, and image.

\* Update existing event details.

\* Delete events.

\* Publish or unpublish events.

\* Manage event categories.

* 1. DATA STRUCTURE OF ALL MODULES:

We have organized one database for the Event Management System design. It can be accessed directly or sequentially by registered users. The database determines files, records, fields, and characters. It can be easily controlled and updated. “Event Management System”

* 1. PROCEDURAL DESIGN:

Process logic (flowchart ) of each module

## User Panel Design

In user panel design, we have done our task for the user. Here we provide facilities for event browsing and registration. In the index page, a user can select any options which are needed by him/her. By selecting options, he/she can see the desired page. Then he/she can get all the oriented information finally. The design of the user panel is shown in the following flowchart…

Browse Events

Select Event Events

View Details

Register/Login Events

Registration

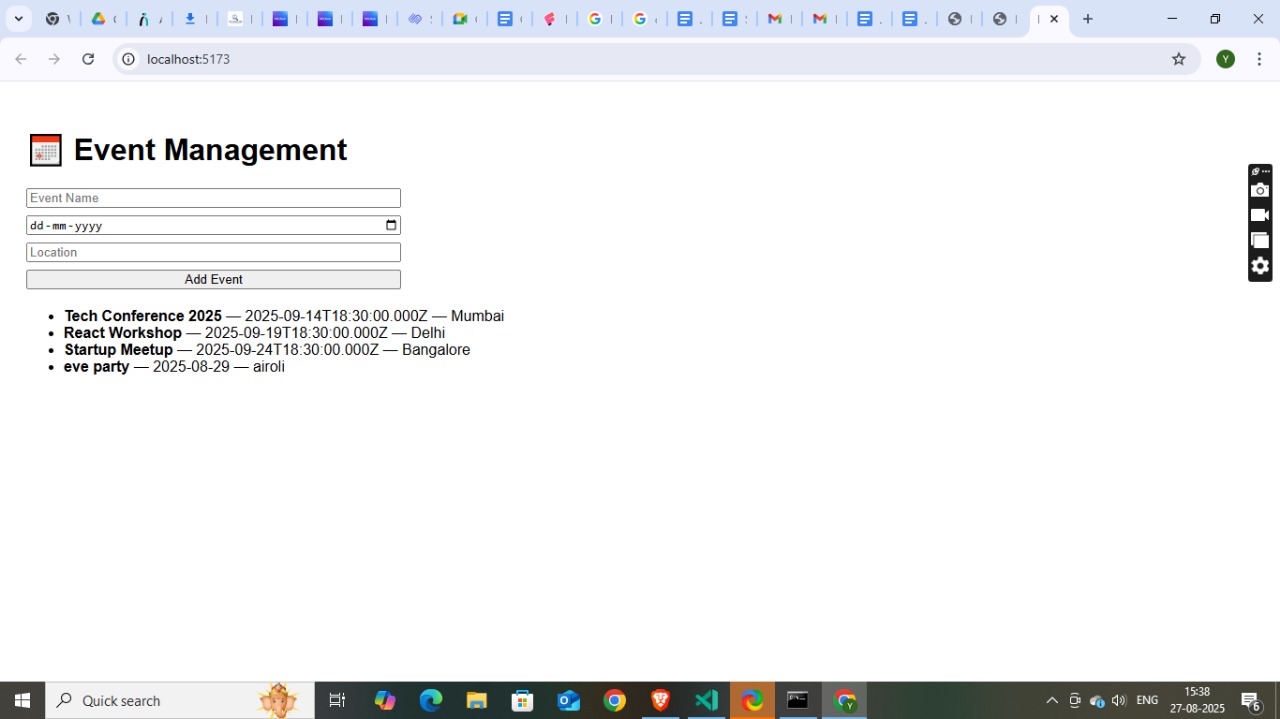
**Fig. 4.1: The user panel flowchart part.**

## Admin Panel Design

We have designed a user login facility to manage and update all of the information. It is a fully secured page. Without appropriate username and password, it cannot be accessed by anyone. For admin login, after giving username and password, we need to click a login button. When we click the login button, it does not directly enter the home page; it stays on the login page. Then it starts a session and sets two variables called username and password. If the username and password are matched with the database, it can enter the home page. It is not possible without clicking the login button. In case if the username or password are not matched with the database, then an "Invalid username or password" message is shown. We can describe the login facility in admin login by using the below flowchart given below —

**SCREENSHOTS**

**Event search**

****

Coding

React Component (EventList.js)\*\*

```jsx

// src/components/EventList.js

import React, { useState, useEffect } from 'react';

import EventCard from './EventCard';

import axios from 'axios'; // Assuming you use axios for API calls

const EventList = () => {

const [events, setEvents] = useState([]);

const [loading, setLoading] = useState(true);

const [error, setError] = useState(null);

useEffect(() => {

const fetchEvents = async () => {

try {

const response = await axios.get('/api/events'); // Your backend API endpoint

setEvents(response.data);

} catch (err) {

setError('Failed to fetch events.');

console.error(err);

} finally {

setLoading(false);

}

};

fetchEvents();

}, []);

if (loading) return <p>Loading events...</p>;

if (error) return <p style={{ color: 'red' }}>{error}</p>;

if (events.length === 0) return <p>No events found.</p>;

return (

<div className="event-list">

{events.map(event => (

<EventCard key={event.\_id} event={event} />

))}

</div>

);

};

export default EventList;

```

Testing

\*\*Unit Testing:\*\* Unit testing where individual program units or object classes are tested. Here by using this testing, we have focused on testing the functionality of methods and React components in isolation.

\*\*Module Testing:\*\* Where this is the combination of unit programs is called a module. Here we tested the unit program (e.g., a group of related React components or a set of API endpoints) where the module programs have dependency.

\*\*Sub-system Testing:\*\* Then we combined some modules for the Preliminary System Testing in our Project (e.g., testing the frontend-backend integration for event creation).

\*\*System Testing:\*\* Where it is the combination of two or more sub-systems and then it is tested. Here we tested the Entire system as per the requirements (e.g., a full user flow from browsing to registration, or an admin flow from login to event management).

\*\*Acceptance Testing:\*\* Normally this type of testing is done to verify if the system meets the customer specified requirements. After submitting this project to User then they tested it and to determine whether to accept the application. It is the system testing performed by the customer(s) to determine whether they should accept the delivery of the system

# 6. Future scope

The Event Management Website can be enhanced with the following features:

\* \*\*Payment Gateway Integration\*\* for paid events and ticket sales.

\* \*\*Calendar Integration\*\* (e.g., Google Calendar, Outlook) for users to add events directly.

\* \*\*Push Notifications\*\* for real-time event updates and reminders.

\* \*\*User Reviews and Ratings\*\* for events.

\* \*\*Live Streaming Integration\*\* for virtual or hybrid events.

\* \*\*Advanced Search & Filters\*\* for refined event discovery (e.g., by date range, price).

\* \*\*AI-driven Event Recommendations\*\* based on user preferences and past attendance.

\* \*\*Mobile App\*\* for better accessibility and native features.

\* \*\*User-Generated Content\*\* submission for events (with admin approval).

\* \*\*Admin Analytics Dashboard\*\* to track event performance, attendance rates, and user engagement.

**Conclusion**

The Event Management Website System offers a comprehensive and efficient solution for managing and delivering event content in a digital format. By automating the process, it enhances productivity, ensures timely updates, and provides a user-friendly interface for both attendees and administrators. The system incorporates secure authentication and role-based access, ensuring appropriate functionality for users and admins. With features like categorized events, registration management, and user tracking, it fosters better user engagement and streamlined event logistics. Designed with scalability and flexibility in mind, the system is well-equipped to adapt to future enhancements, making it a reliable and robust platform for modern event management.

# BIBLIOGRAPHY

* *Web Development with Django* by Ben Shaw & Saurabh
* *Django 4 By Example* by Antonio Mele

# REFERENCES

\* React Official Documentation – `https://react.dev/`

\* Node.js Official Website – `https://nodejs.org/` (if using Node.js backend)

\* Express.js Documentation – `https://expressjs.com/` (if using Express.js backend)

\* [Your chosen database] Documentation – `[Link to your database documentation]`

\* MDN Web Docs – `https://developer.mozilla.org/`

\* Stack Overflow – `https://stackoverflow.com/`