

A Field Project Report on

Eventify-Smart Event Management System

Submitted

In partial fulfillment of the requirements for the award of the degree

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE and ENGINEERING

By

Syed Tahir Ali	(231FA04750)
Polisetty Yoga Rajitha	(231FA04774)
Rachagarla Swathi	(231FA04842)
Davu Gnanendra Kumar	(231FA04845)

Under the Guidance of
Mr.T.Narasimha Rao
Assistant Professor, CSE



VIGNAN'S

FOUNDATION FOR SCIENCE, TECHNOLOGY & RESEARCH

(Deemed to be University) - Estd. u/s 3 of UGC Act 1956

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SCHOOL OF COMPUTING AND INFORMATICS

VIGNAN'S FOUNDATION FOR SCIENCE, TECHNOLOGY & RESEARCH
(Deemed to be University)
Vadlamudi, Guntur -522213, INDIA.

April, 2025



VIGNAN'S


FOUNDATION FOR SCIENCE, TECHNOLOGY & RESEARCH

(Deemed to be University) - Estd. u/s 3 of UGC Act 1956

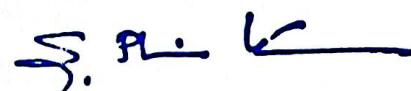
CERTIFICATE

This is to certify that the field project entitled "*EVENTIFY-SMART EVENT MANAGEMENT SYSTEM*" is being submitted by [Syed Tahir Ali], [231FA04750], [Polishetty Yoga Rajitha], [231FA04774], [Rachagarla Swathi], [231FA04842], and [Davu Gnanendra Kumar], [231FA04845] in partial fulfilment of the requirements for the degree of **Bachelor of Technology (B.Tech.) in Computer Science and Engineering** at Vignan's Foundation for Science, Technology and Research (Deemed to be University), Vadlamudi, Guntur District, Andhra Pradesh, India.

This is a bonafide work carried out by the aforementioned students under my guidance and supervision.



Guide



HoD, CSE



Project Review Committee

HoD
Dept. of Computer Science & Engineer.
VFSTR Deemed to be University
VADLAMUDI - 522 213
Guntur Dist. A.P. India.

DECLARATION**Date:26-04-2025**

We hereby declare that the work presented in the field project titled "EVENTIFY-SMART EVENT MANAGEMENT SYSTEM" is the result of our own efforts and investigations.

This project is being submitted under the supervision of **Mr.T.Narasimha Rao, Assistant Professor,CSE** in partial fulfillment of the requirements for the Bachelor of Technology (B.Tech.) degree in Computer Science and Engineering at Vignan's Foundation for Science, Technology and Research (Deemed to be University), Vadlamudi, Guntur, Andhra Pradesh, India.

Syed Tahir Ali

(231FA04750)

Signature *Tahir Ali*

Polisetty Yoga Rajitha

(231FA04774)

Signature *P.Yoga Rajitha*

Rachagarla Swathi

(231FA04842)

Signature *R.Swathi*

Davu Gnanendra Kumar

(231FA04845)

Signature *D.Gnanendra Kumar*

Contents

Chapter No.		Description	Page No.
1		Introduction	5
	1.1	Problem definitions	5
	1.2	Existing system	5
	1.3	Proposed system	6
	1.4	Literature review	6
2		System Requirements	7
	2.1	Hardware & Software requirements	8
	2.2	Software requirements Specification(SRS)	8-9
3		System Design	10-11
	3.1	Modules of system	11
	3.2	UML Diagrams	11-14
4		Implementation	15
	4.1	Sample code	15-37
	4.2	Test Cases	38
5		Results	39
	5.1	Output Screens	40-42
6		Conclusion	43
		References	43

“SMART EVENT MANAGEMENT SYSTEM”

1.INTRODUCTION

Event planning can be a complex and time-consuming process, requiring careful coordination of multiple elements such as budgeting, guest management, food preferences, and décor. To streamline this experience, Eventify provides a cutting-edge, web-based event management system that simplifies event planning for individuals and organizations alike. By offering a user-friendly interface with dynamic customization options, Eventify ensures that users can effortlessly organize weddings, birthdays, corporate gatherings, and more while staying within their desired budget. This innovative platform enhances transparency and efficiency, making event planning more accessible and stress-free for hosts and planners.

1.1 Problem Definition

Event planners often face challenges in managing multiple aspects of an event, including budgeting, food preferences, décor, and guest numbers. Traditional manual methods such as spreadsheets or handwritten notes can be time-consuming, error-prone, and inefficient. Additionally, planners must manually communicate their preferences to vendors, which may result in miscommunication or delays. The lack of a centralized tool for managing all these details complicates the planning process, making it difficult to track and adjust the budget and event requirements effectively.

1.2 Existing System

1. **Manual Planning:** Event organizers often rely on spreadsheets or handwritten notes to plan and manage event details.
2. **Budget Estimation:** Manually calculating costs, tracking spending, and adjusting budget allocations is error-prone and time-consuming.
3. **Limited Customization:** Event planners must manually communicate their preferences to vendors without a centralized tool for quick changes.

1.3 Proposed System

1. **Centralized Dashboard:** Users can manage all aspects of their event in one place — from type of event to budget, food choices, and décor preferences.
2. **Dynamic Cost Calculator:** As users make selections for décor, food, and guest numbers, the system updates the cost instantly.
3. **Event Customization:** The platform allows full customization, ensuring the event fits the user's budget while offering flexibility in choices.
4. **Enhanced User Experience:** With an easy-to-navigate interface, users can manage multiple events simultaneously, track their budget, and visualize total costs in real time.

1.4 Literature Review

Event management has evolved significantly with the rise of digital solutions, making planning more efficient and accessible. Several web-based event management platforms have been developed to simplify the process, offering features like budget tracking, customization, and guest management. This literature survey explores existing event management systems, their strengths, limitations, and how Eventify aims to improve upon them.

2.SYSTEM REQUIREMENTS

Here are the shortened system requirements for the Event Management System (Eventify):

1. Functional Requirements:

- Pages: Homepage, Event Selection, Decor, Food, Cart, Confirmation.
- Event Selection: Choose event type and store selection in local storage.
- Decor Selection: Pick theme, calculate cost.
- Food Selection: Choose food type, menu, number of guests, calculate cost.
- Cart: Display selections, calculate total cost.
- Payment: Validate card details (name, number, CVV).
- Confirmation: Display booking confirmation, clear data.

2. Non-Functional Requirements:

- Performance: Page loads and actions under 2 seconds.
- Scalability: Handle up to 1000 users.
- Usability: Clear UI, easy navigation, mobile responsive.
- Compatibility: Modern browsers (Chrome, Firefox, Safari, Edge).
- Security: Use HTTPS for payment, validate card details frontend.

3. Technical Requirements:

- Frontend: HTML5, CSS3, JavaScript.
- Data Storage: Use localStorage for temporary data.
- No backend (optional in real-world scenarios).

4. Dependencies:

- Browser: Compatible with modern browsers.
- Internet: Required for accessing the system and payment gateway.

5. Limitations:

- **Local Storage:** Limited to 5MB for data storage.
- **Mock Payment:** Real payment integration needed for production.

2.1 Hardware & Software requirements

Hardware Requirements	Software Requirements
Computer/Server: <ul style="list-style-type: none"> • Processor: Intel i3 or equivalent • RAM: 4GB (8GB recommended) • Storage: 1GB free space • Internet: Stable connection (2Mbps) 	Web Browsers: <ul style="list-style-type: none"> • Chrome (latest version) • Firefox (latest version) • Edge (latest version) • Safari (latest version)
Web Hosting: <ul style="list-style-type: none"> • Server with PHP or Node.js support • Example platforms: AWS, DigitalOcean • Ability to handle dynamic requests • Supports HTTPS for security 	Operating System: <ul style="list-style-type: none"> • Windows 10+ or newer • macOS 10.14+ or newer • Linux (latest version) • Supports modern web standards
Server for Production: <ul style="list-style-type: none"> • Dedicated or cloud server with sufficient resources (CPU, RAM, and storage) to handle live traffic • Example: 2-4 CPU cores, 8GB RAM, 50GB+ storage for small to medium-scale applications 	Development Tools: <ul style="list-style-type: none"> • Editor/IDE: VS Code, Sublime, or Atom • Version Control: Git, GitHub/Bitbucket • Terminal/Command Line tools • Dependency management (e.g., npm for Node.js)

2.2 Software Requirements Specifications (SRS)

1. Frontend Requirements:

- **Browsers:** Chrome, Firefox, Edge, Safari (latest versions).
- **Technologies:** HTML5, CSS3, JavaScript (ES6+).

- **Libraries/Tools:** Google Fonts (Poppins), custom JavaScript for interactivity.

2. Backend (Optional):

- **Languages:** Node.js with Express or PHP for server-side logic.
- **Database:** MySQL or MongoDB (optional).

3. Payment Gateway:

- **APIs:** Stripe or PayPal for secure payment processing.

4. Security:

- **SSL/HTTPS** for secure connections.
- **Form Validation:** Client-side (JavaScript) and server-side.

5. Development Tools:

- **IDE/Text Editor:** Visual Studio Code, Sublime Text, or Atom.
- **Version Control:** Git with GitHub, GitLab, or Bitbucket.

6. Hosting/Server:

- **Web Server:** Apache or Nginx.
- **Cloud Hosting:** AWS, Digital Ocean (optional).

7. Testing:

- **Cross-Browser & Responsive Testing.**
- **Unit/Integration Testing** (Optional for backend).

This setup ensures smooth operation, security, and responsiveness across devices.

3. SYSTEM DESIGN

1. Frontend (UI):

- **Pages:**
 - **Home:** Event selection (Wedding, Birthday, Corporate).
 - **Decor:** Choose decoration themes.
 - **Food:** Select food type (Veg, Non-Veg), guest count.
 - **Cart:** Show selections, total cost, and payment.
 - **Confirmation:** Booking confirmation.
- **Technologies:** HTML5, CSS3, JavaScript (local Storage for data persistence).

2. Backend (Optional):

- **Server:** Node.js (Express) or PHP for API handling.
- **Database:** MongoDB/MySQL to store selections and user data.
- **Payment:** Stripe/PayPal for secure transactions.

3. Data Flow:

1. **Event Selection** → Decor Page → Food Page → Cart → Payment.
2. **Local Storage** stores user choices across pages.

4. Security & Performance:

- **SSL/TLS** encryption for secure transactions.
- **Input Validation** on both client and server sides.
- **Responsive Design** and optimized images for performance.

5. Technologies:

- **Frontend:** HTML, CSS, JavaScript.

- **Backend:** Node.js/PHP, MongoDB/MySQL (optional).
- **Payment:** Stripe/PayPal API.

3.1 Modules of the Eventify System

1. **Event Selection:** Users choose an event type (Wedding, Birthday, Corporate), view details with pricing, and store selections in localStorage.
2. **Decoration Selection:** Users pick a decor theme (Royal, Floral, Modern), and the choice is stored for the next step.
3. **Food Selection:** Users select a food type (Veg/Non-Veg), view a dynamic menu, input guest count, and save the selection in localStorage.
4. **Cart Summary:** Displays event, decor, food, and guest count details, calculates total cost, and provides payment options.
5. **Payment Processing:** Users enter card details (name, number, CVV) and complete payment via Stripe or PayPal.
6. **Booking Confirmation:** Displays a confirmation message, clears localStorage, and redirects users to the confirmation page.

3.2 UML Diagram

A **use case diagram** shows the interaction between users (actors) and the system.

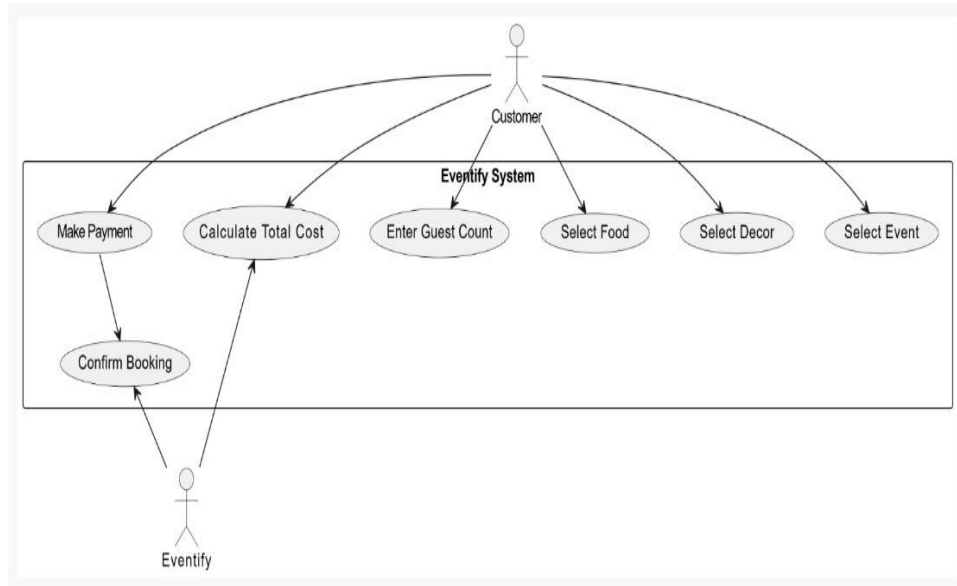
Actors:

1. **User (Customer)** – Selects event type, decor, food, and makes a booking.
2. **System (Eventify)** – Processes selections, calculates total cost, and handles booking.

Use Cases:

- **Select Event** – User selects an event type (Wedding, Birthday, Corporate).
- **Select Decor** – User chooses decoration themes.
- **Select Food** – User selects food type and menu.
- **Enter Guest Count** – User inputs the number of guests.

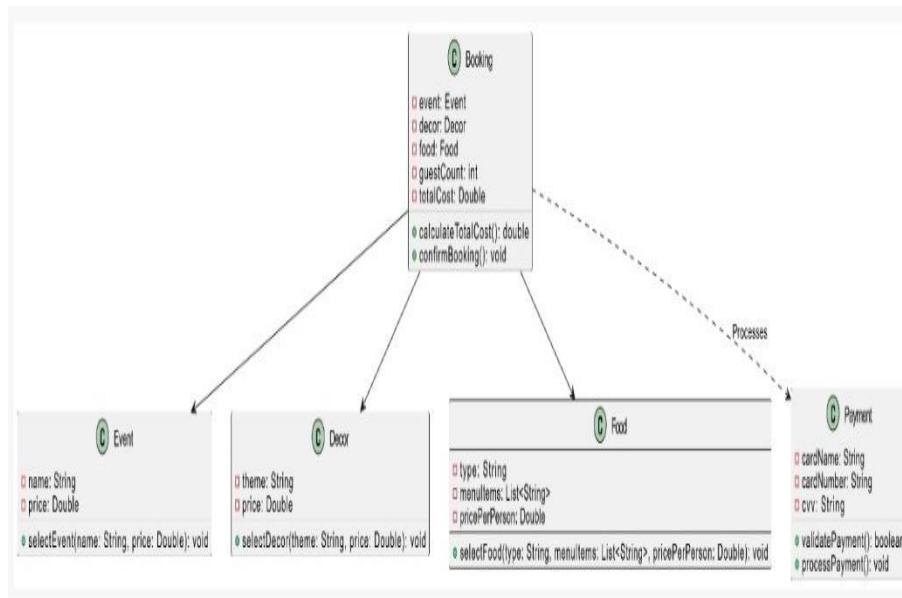
- **Calculate Total Cost** – System computes event price based on selections.
- **Make Payment** – User enters payment details and confirms booking.
- **Confirm Booking** – System generates a booking confirmation.



Class Diagram

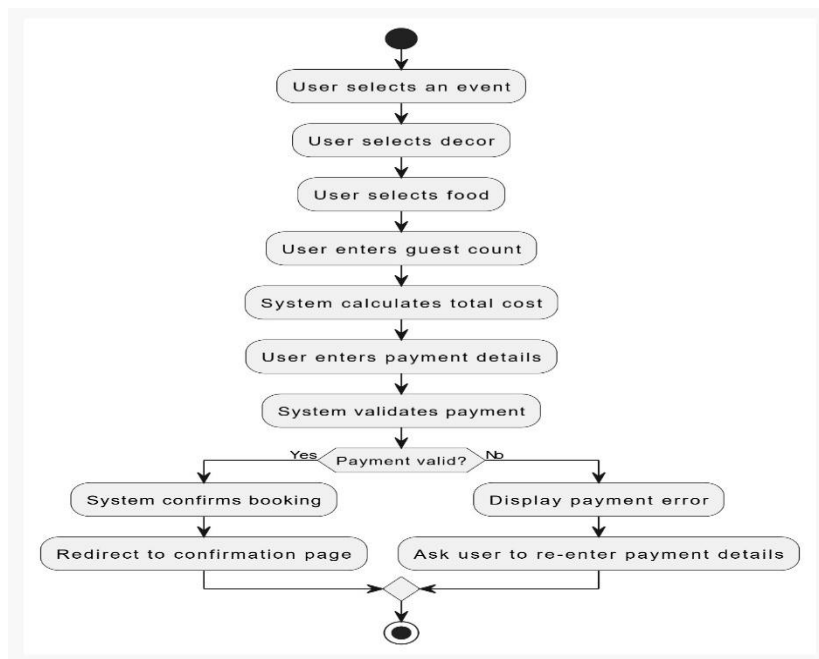
This class diagram represents the structure of the Eventify System, which facilitates event booking. The main classes are:

1. **Event**: Represents different event types with a name and price.
2. **Decor**: Stores decoration themes and their prices.
3. **Food**: Manages food selection with type, menu items, and cost per person.
4. **Booking**: Aggregates event, decor, and food details; calculates total cost and confirms the booking.
5. **Payment**: Handles payment details and validation.



Textual Representation of the Activity Diagram

1. User selects an event type.
2. System stores the selection and navigates to the decor page.
3. User selects a decor theme.
4. System stores the decor selection and navigates to the food selection page.
5. User selects a food type.
6. System displays menu options.
7. User enters the guest count.
8. System calculates the total cost.
9. User enters payment details.
10. System validates and processes the payment.
11. System confirms the booking and redirects the user to the confirmation page.



4. IMPLEMENTATION

1. HTML Pages: index.html, decor.html, food.html, cart.html, and confirmation/.html manage event selection, decoration, food options, cart summary, and booking confirmation, using navigation based on user choices.
2. JavaScript: Handles event selection and stores user choices in localStorage, with functions to navigate between pages and display appropriate content based on user selections (e.g., show food menu, calculate total).
3. Cart and Payment: On the cart.html page, users can review selections and enter payment details, which are validated before proceeding with the booking.
4. CSS Styling: The design is enhanced with CSS for a smooth, responsive layout, with flexbox for positioning and hover effects for interactive elements.

4.1 Sample code

1.index.html

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Welcome to Eventify</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<header>

<h1> EVENTIFY </h1>

<p>Smart Event Management System....!</p>
```

```
</header>

<section class="event-selection">

<h2 class="event-title">Select Your Event Type</h2>

<div class="event-options">

<div class="event" onclick="selectOption(this, 'event', 'Wedding', 400)">



<p>Wedding - $500</p>

</div>

<div class="event" onclick="selectOption(this, 'event', 'Birthday Party', 300)">



<p>Birthday Party - $300</p>

</div>

<div class="event" onclick="selectOption(this, 'event', 'Corporate Event', 700)">



<p>Corporate Event - $700</p>

</div>

</div>

</section>

<footer>

<h3>Contact Us</h3>

<p>Email: support@eventify.com</p>

<p>Phone: +123 456 7890</p>

</footer>

<script src="eventify_script.js"></script>

</body>
```

</html>

2.decor.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Select Decor - Eventify</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<header>

<h1>Select Your Event Decoration </h1>

</header>

<section class="decor-selection">

<h2>Choose a Decor Theme</h2>

<div class="decor-options">

<div class="decor" onclick="selectOption(this, 'decor', 'Royal Theme', 70)">

<p>Royal Theme - \$200</p>

</div>

<div class="decor" onclick="selectOption(this, 'decor', 'Floral Theme', 70)">

<p>Floral Theme - \$150</p>

```

</div>

<div class="decor" onclick="selectOption(this, 'decor', 'Modern Theme', 70)">



<p>Modern Theme - $180</p>

</div>

</div>

</section>

<script src="eventify_script.js"></script>

</body>

</html>

```

3.food.html

```

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Eventify - Select Food</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<h1>Select Food Type</h1>

<div class="food-options">

<div class="food" onclick="selectOption(this, 'food', 'Veg', 300)">



<p>Veg - $300 per person</p>

```

```

</div>

<div class="food" onclick="selectOption(this, 'food', 'Non-Veg', 500)">



<p>Non-Veg - $500 per person</p>

</div>

</div>

<h2>Menu</h2>

<div id="menu-items"></div>

<label>Number of Guests: <input type="number" id="guestCount"></label>

<<button id="next-button" onclick="goToCart()">PROCEED</button>

<script src="eventify_script.js"></script>

</body>

</html>

```

4.cart.html

```

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Cart - Eventify</title>

<link rel="stylesheet" href="styles.css">

</head>

<body onload="loadCart()">

<header>

<h1>Your Event Cart </h1>

```

```
</header>

<section>

<div id="cart-details"></div>

</section>

</body>

</html>
```

5.confirmation.html

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Booking Confirmed - Eventify</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<header>

<h1> Booking Confirmed!</h1>

</header>

<section>

<p>Thank you for choosing us! Your event has been successfully booked.</p>

<a href="index.html">Go to Home</a>

</section>

<script>

window.onload = function() {
```



```
console.log("Confirmation page loaded.");
```

```
};
```

```
</script>
```

```
</body>
```

```
</html>
```

6.styles.css

```
/* Importing Google Font */
```

```
@import
```

```
url('https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600&display=swap');
```

```
/* General Body Styles */
```

```
body {
```

```
font-family: 'Poppins', Arial, sans-serif;
```

```
background-image: url('images/background1.jpg');
```

```
background-size: cover;
```

```
background-attachment: fixed;
```

```
background-position: center;
```

```
color: #ffffff;
```

```
text-align: center;
```

```
margin: 0;
```

```
padding: 0;
```

```
min-height: 100vh;
```

```
display: flex;
```

```
flex-direction: column;
```

```
justify-content: center;
```

```
}
```

```
/* Header Section */

header {

padding: 20px;

background: rgba(0, 0, 0, 0.6);

border-radius: 10px;

margin: 20px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.3);

}
```

```
header h1 {

color: #ffcc00;

text-shadow: 2px 4px #000;

font-size: 48px;

margin-bottom: 10px;

animation: fadeInDown 1s ease-in-out;

}
```

```
header p {

color: #ffdd55;

font-size: 20px;

}
```

```
/* Event Section */

.event-selection h2 {

color: #ff8800;

text-shadow: 2px 4px #000;
```

```
font-size: 36px;

margin-bottom: 20px;

}

.event-selection {

text-align: center;

margin-bottom: 20px;

}


.event-options {

display: flex;

justify-content: center;

gap: 20px;

}

.event {

display: inline-block;

flex-shrink: 0;

width: 250px;

text-align: center;

transition: transform 0.3s;

}

.event:hover {

transform: scale(1.05);

}

.event img {

width: 250px;
```

```

height: 300px;

object-fit: cover;

border-radius: 10px;

box-shadow: 0 4px 8px rgba(255, 255, 255, 0.2);

}

.event p {

color: #ffdd55;

font-size: 18px;

font-weight: bold;

}

/* Footer Section */

footer {

background: rgba(0, 0, 0, 0.6);

border-radius: 10px;

padding: 15px;

margin-top: 30px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.3);

}

footer h3 {

color: #ffcc00;

text-shadow: 2px 4px #000;

font-size: 24px;

margin-bottom: 5px;

}

footer p {

```

```

color: #ffdd55;

}

/* Decor Section */

.decor-selection h2 {

color: #ff8800;

font-weight: bold;

}

.decor-options {

display: flex;

justify-content: center;

flex-wrap: wrap;

gap: 20px;

}

.decor-options .decor {

border-radius: 10px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);

overflow: hidden;

transition: transform 0.3s;

width: 250px;

text-align: center;

}

.decor-options .decor:hover {

transform: scale(1.05);

}

.decor-options img {

```

```

width: 100%;

height: 200px;

object-fit: cover;

border-bottom: 2px solid #ff8800;

border-radius: 10px 0 0;

}

.decor-options p {

padding: 10px;

margin: 0;

color: #ff5500;

font-weight: bold;

}

/* Food Section */

.food-options {

display: flex;

justify-content: center;

gap: 40px;

margin-top: 80px;

}

.food {

background: rgba(255, 255, 255, 0.8);

border-radius: 10px;

transition: transform 0.3s;

width: 300px;

text-align: center;

```



```

color: #ff8800;

text-shadow: 2px 4px #000;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.3);

padding: 10px;

}

.food:hover {

transform: scale(1.05);

}

.food img {

width: 100%;

height: 200px;

object-fit: cover;

border-radius: 10px;

}


.food p {

padding: 10px;

margin: 0;

font-weight: bold;

color: #ff6600;

}

/* Confirmation Section */

section p {

font-size: 1.5em;

color: #f40b9b;

```

```

margin-bottom: 30px;

animation: fadeIn 2s ease-in-out;

}

section a {

color: #eaf4f5;

text-decoration: none;

font-weight: bold;

background-color: rgba(0, 193, 212, 0.2);

padding: 10px 20px;

border-radius: 5px;

transition: background-color 0.3s, color 0.3s;

}

section a:hover {

background-color: #00C1D4;

color: #ffffff;

}

/* Cart Details */

#cart-details {

background: rgba(0, 0, 0, 0.7);

border-radius: 15px;

padding: 30px;

width: 400px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.3);

}

#cart-details p {

```

```
line-height: 1.6;

font-size: 18px;

margin-bottom: 10px;

}

#cart-details h3 {

font-size: 24px;

color: #ffcc00;

text-align: center;

margin-bottom: 20px;

}

#cart-details h2 {

text-align: center;

color: #ff8800;

margin-top: 20px;

}

input[type="text"] {

width: 100%;

padding: 10px;

margin: 10px 0;

border-radius: 5px;

border: none;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.2);

text-align: center;

}

button {
```

```
width: 100%;

padding: 10px;

background: #ff6600;

color: #fff;

border: none;

border-radius: 5px;

font-weight: bold;

cursor: pointer;

transition: background 0.3s;

text-align: center;

}

button:hover {

background: #cc5200;

text-align: center;

}

/* Animations */

@keyframes fadeInDown {

0% {

opacity: 0;

transform: translateY(-50px);

}

100% {

opacity: 1;

transform: translateY(0);

}
```

```

@keyframes fadeIn {

0% {

opacity: 0;

}

100% {

opacity: 1;

}

}

#guestCount {

margin-bottom: 10px; /* Gap between input and button */

}

#next-button {

background-color: #87ccf7; /* Change button color */

color: white;

padding: 3px 10px; /* Smaller size */

border: none;

border-radius: 3px;

font-size: 12px;

cursor: pointer;

margin-top: 10px;

}

#next-button:hover {

background-color: #87ccf7; /* Hover color */

}

```

7.eventify_script.js

```

let selections = JSON.parse(localStorage.getItem("selections")) || {

event: null,

decor: null,

food: null,

guestCount: 0,

foodItems: []

};

function selectOption(element, category, name, price) {

let elements = document.querySelectorAll(`${category}`);

elements.forEach(el => el.classList.remove("selected"));

element.classList.add("selected");

selections[category] = { name, price };

localStorage.setItem("selections", JSON.stringify(selections));

if (category === 'event') {

window.location.href = "decor.html";

} else if (category === 'decor') {

window.location.href = "food.html";

} else if (category === 'food') {

showMenu(name);

document.getElementById('menu-section').scrollIntoView({ behavior: 'smooth' });

}

}

function showMenu(type) {

let menuItems = document.getElementById("menu-items");

```



```

menuItems.innerHTML = "";

let items = type === 'Veg' ? ['Paneer Tikka', 'Veg Biryani', 'Dal Makhani' , 'dal' ,
'sambar','panipuri','potatofry','cornsamosa','laddoo'] : ['Chicken Curry', 'Mutton Biryani', 'Fish
Fry','chicken 555','chicken 65','chicken biryani','mutton fry'];

items.forEach(item => {

let itemDiv = document.createElement("div");

itemDiv.className = "menu-item";

itemDiv.innerHTML = `${item}`;

menuItems.appendChild(itemDiv);

});

}

function goToCart() {

selections.guestCount = parseInt(document.getElementById("guestCount").value) || 0;

if (!selections.food) {

alert("Please select a food option.");

return;

}

localStorage.setItem("selections", JSON.stringify(selections));

window.location.href = "cart.html";

}

document.getElementById("cart-details").innerHTML = `

<p><strong>Event:</strong> ${selections.event.name} - $$${selections.event.price}</p>

<p><strong>Decor:</strong> ${selections.decor.name} - $$${selections.decor.price}</p>

<p><strong>Food:</strong>  ${selections.food.name}  -  $$${selections.food.price}  per
person</p>

```

```
<p><strong>Guests:</strong>          ${ selections.guestCount }</p><p><strong>Menu  
Items:</strong> ${ selections.foodItems.join(", ") || "None" }</p>
```

```
<h3>Total Cost: $$ {totalCost}</h3>
```

```
<h2>Payment Details</h2>
```

```
<input type="text" id="card-name" placeholder="Cardholder Name" required pattern="[A-Za-  
z ]+" title="Only letters allowed"><br>
```

```
<input type="text" id="card-number" placeholder="Card Number" required pattern="\d{14}"  
title="Exactly 14 digits required"><br>
```

```
<input type="text" id="cvv" placeholder="CVV" required pattern="\d{3}" title="Exactly 3  
digits required"><br>
```

```
<button onclick="payAndBook()">Pay and Book</button>
```

```
`;  
`;
```

```
function payAndBook() {
```

```
let name = document.getElementById("card-name").value;
```

```
let cardNumber = document.getElementById("card-number").value
```

```
let cvv = document.getElementById("cvv").value;
```

```
// Validation Patterns
```

```
let nameRegex = /^[A-Za-z ]+$/;
```

```
let cardNumberRegex = /^\d{14}$/;
```

```
let cvvRegex = /^\d{3}$/;
```

```
if (!nameRegex.test(name)) {
```

```
alert("Cardholder Name should contain only letters.");
```

```
return;
```

```
}
```

```
if (!cardNumberRegex.test(cardNumber)) {
```

```
alert("Card Number should be exactly 14 digits.");
```

```

return;

}

if (!cvvRegex.test(cvv)) {

alert("CVV should be exactly 3 digits.");

return;

}

localStorage.clear();

window.location.href = "confirmation.html";

}

```

8.Backend_eventify:

Commands:

```
mkdir backend_eventify
```

```
cd backend_eventify
```

```
npm init -y
```

```

{

  "name": "backend_eventify",

  "version": "1.0.0",

  "main": "index.js",

  "scripts": {

    "test": "echo \"Error: no test specified\" && exit 1"

  },

  "keywords": [],

  "author": "",

  "license": "ISC",

  "description": "",

```

```
"dependencies": {  
  "cors": "^2.8.5",  
  "express": "^5.1.0"  
}  
}
```

9.Server.js

```
// server.js  
  
const express = require('express');  
  
const cors = require('cors');  
  
const app = express();  
  
const PORT = 3000;  
  
  
app.use(cors());  
app.use(express.json());  
  
  
app.post("/api/book", (req, res) => {  
  const booking = req.body;  
  
  console.log("📄 New Booking Received:", booking);  
  
  
  // Optional: Generate a random booking ID  
  
  const bookingId = "EVT" + Math.floor(100000 + Math.random() * 900000);  
  
  
  // Respond with confirmation message  
  
  res.json({  
    success: true,
```

```
message: "🔒 Booking confirmed! Thank you, " + booking.payment.name + ".",  
  
    bookingId: bookingId  
  
});  
  
});  
  
app.listen(PORT, () => {  
  
    console.log(☒ Backend server running at http://localhost:${PORT}`);  
  
});
```

```
C:\Users\racha\OneDrive\Desktop\final event\backend_eventify>node server.js
```

```
Backend server running at http://localhost:3000
```

```
^C
```

```
C:\Users\racha\OneDrive\Desktop\final event\backend_eventify>
```

```
C:\Users\racha\OneDrive\Desktop\final event\backend_eventify>node server.js
```

```
☒ Backend server running at http://localhost:3000
```

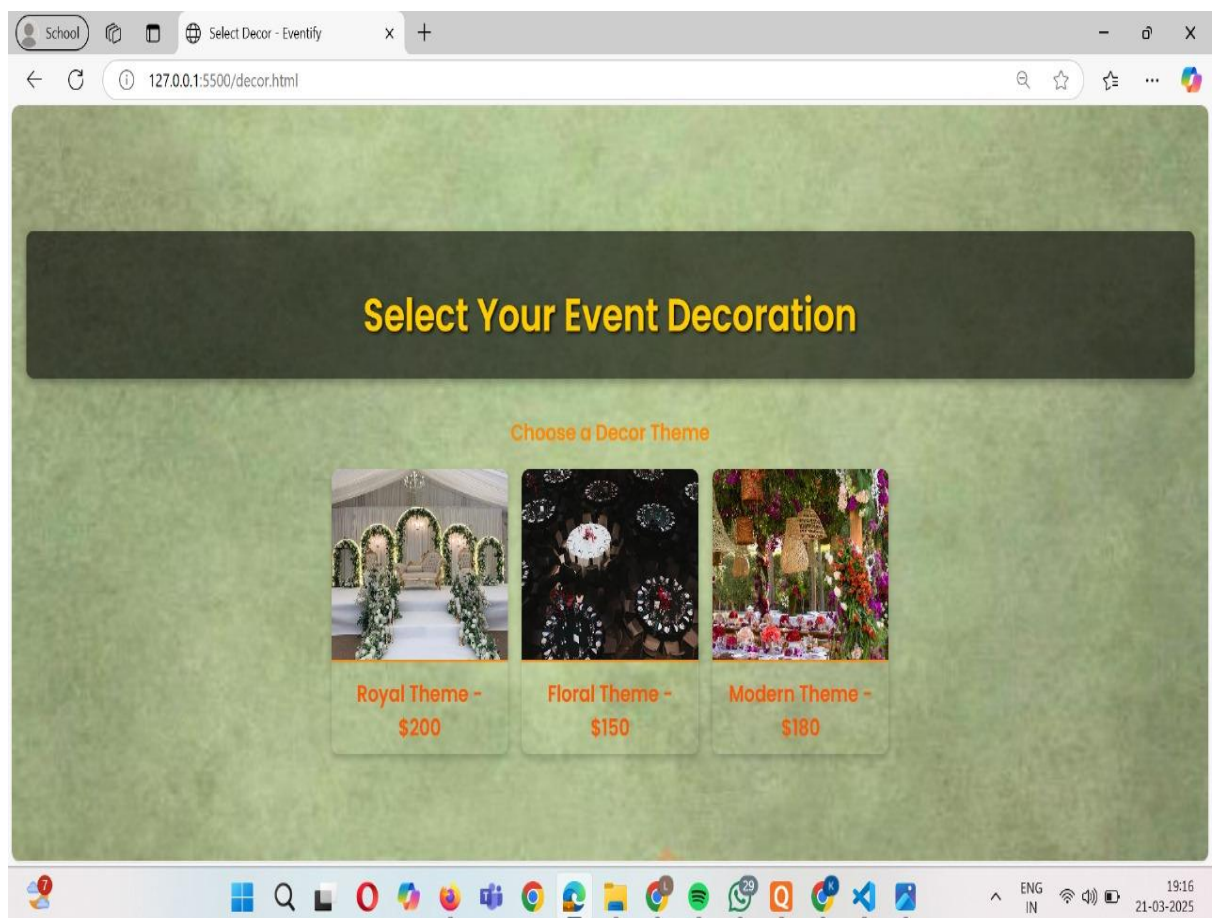
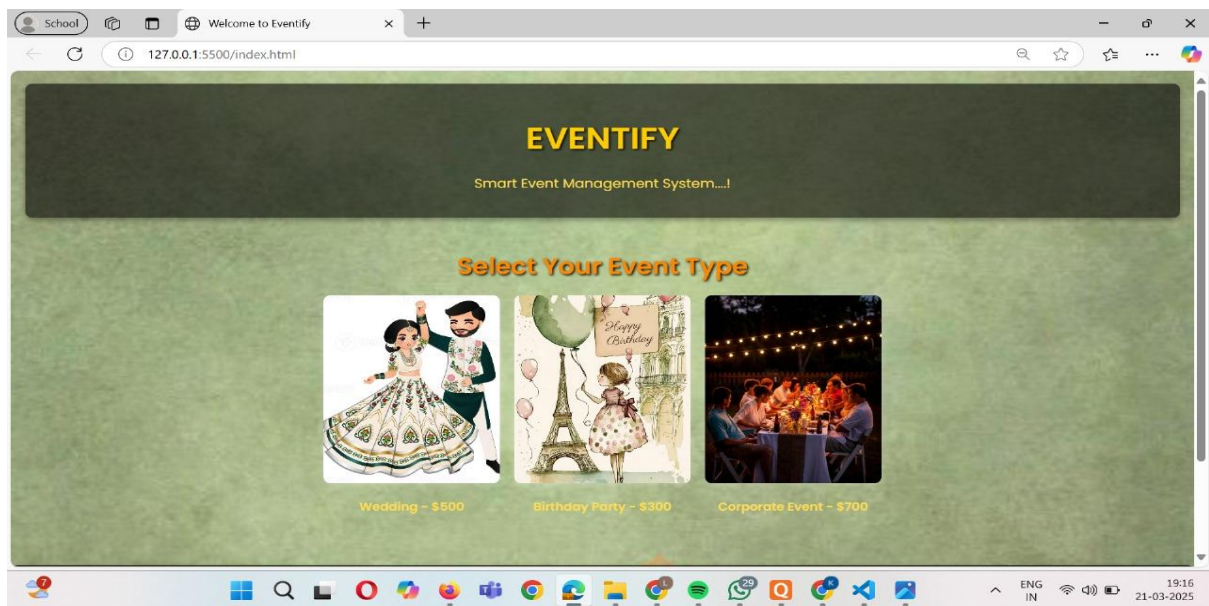
4.2 Test cases

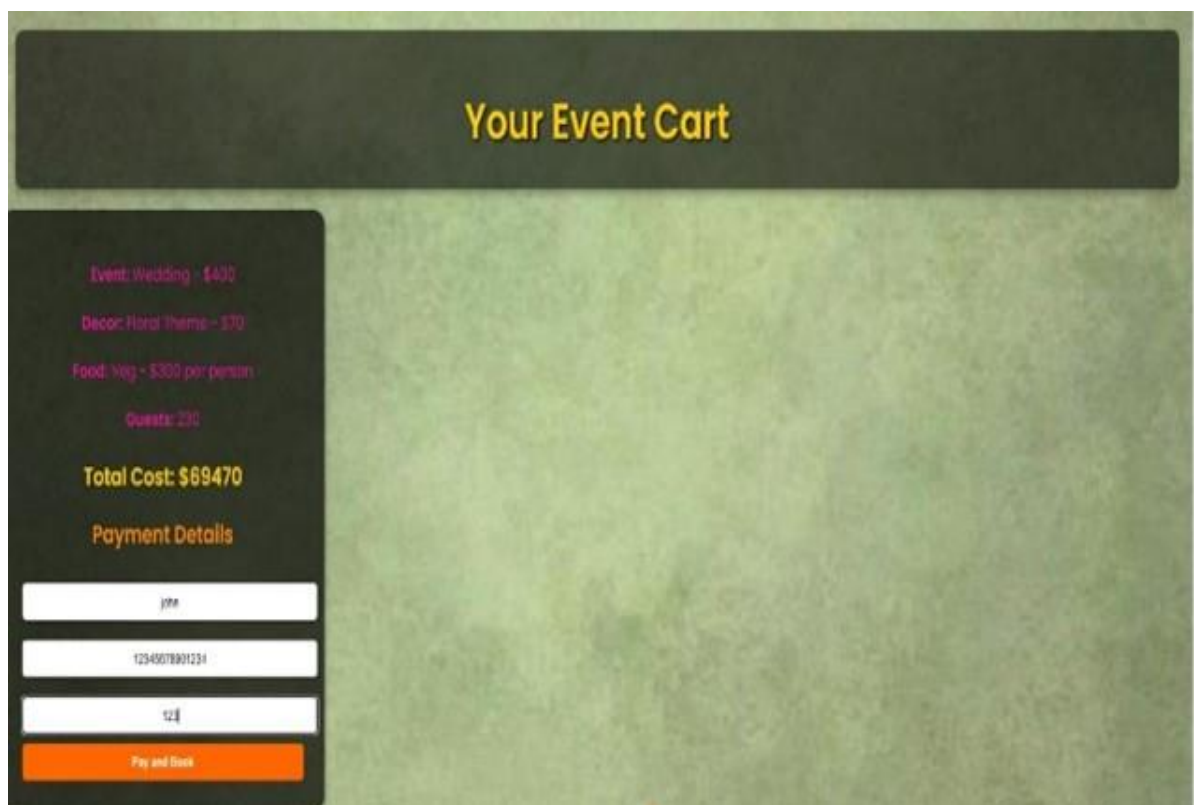
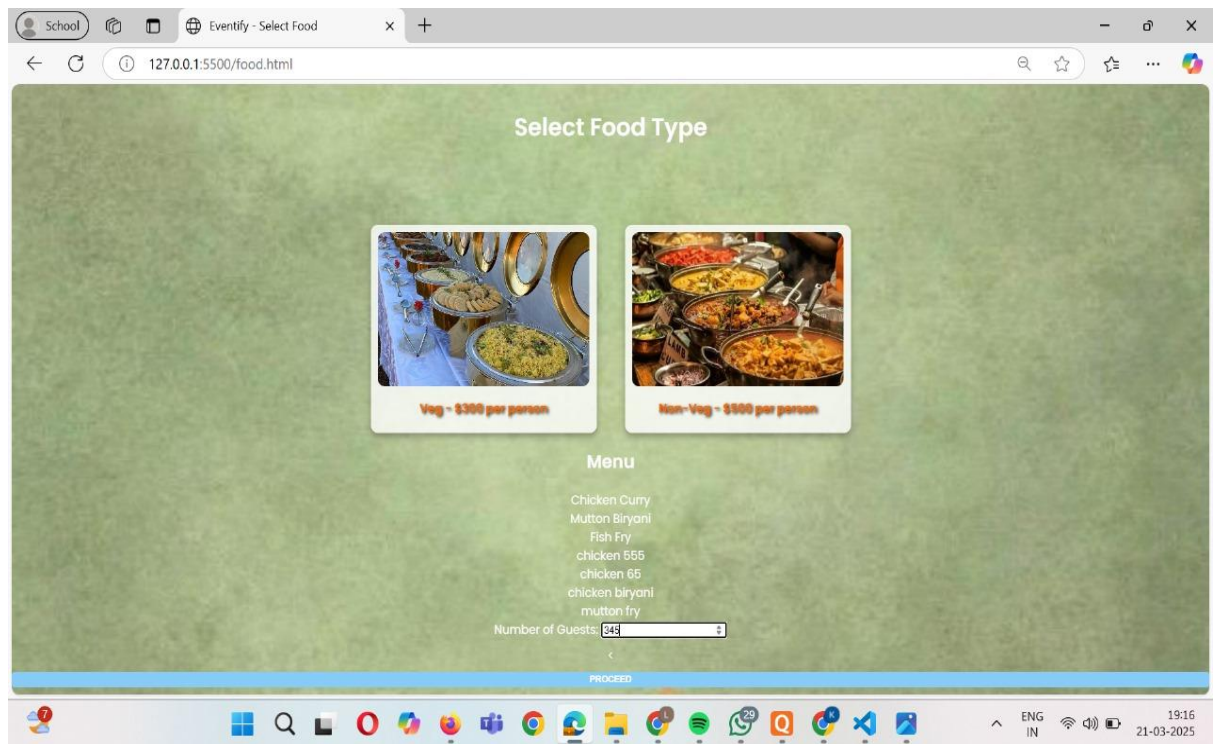
Test Case ID	Test Case Name	Test Steps	Expected Result
TC-1	Event Selection	Click on an event on index.html.	Event is selected and redirects to decor.html.
TC-2	Decor Selection	Select a decor theme on decor.html.	Decor is selected and redirects to food.html.
TC-3	Food Selection	Select food type and enter guest count.	Food and guest count are selected.
TC-4	Cart Review	Click "PROCEED" to view cart on cart.html.	Cart displays selected event, decor, food, and total cost.
TC-5	Payment Validation	Enter valid/invalid card details on cart.html.	Validation errors for invalid input; success redirects to confirmation.html.
TC-6	Confirmation	After booking, check confirmation.html.	Confirmation message is displayed.
TC-7	LocalStorage Check	After selecting event, decor, food, and guests.	Data is saved in local Storage.
TC-8	UI Responsiveness	Check the layout on mobile/responsive view.	UI adjusts to small screens.

5. RESULTS

1. **Event Selection:** Clicking an event type (Wedding, Birthday, Corporate) marks it as selected, saves details in local Storage, and redirects to the decor page.
2. **Decor Selection:** Choosing a decor theme (Royal, Floral, Modern) saves details in localStorage and redirects to the food selection page.
3. **Food Selection:** Selecting food type (Veg/Non-Veg) displays the relevant menu, allows guest count input, saves details in localStorage, and redirects to the cart page.
4. **Cart Review:** The cart displays selected event, decor, food type, guest count, and total cost, with an option to proceed to payment.
5. **Payment Validation:** Valid payment details (cardholder name, number, CVV) allow redirection to the confirmation page; errors prompt user correction.
6. **Booking Confirmation:** After successful payment, a "Booking Confirmed!" message appears with a "Go to Home" link to return to the homepage.
7. **LocalStorage Data Handling:** User selections are saved across pages and persist until payment is completed, after which data is cleared.
8. **Navigation Flow:** Each step follows a logical sequence, ensuring smooth transitions between event, decor, food, cart, payment, and confirmation pages.
9. **Error Handling:** Invalid user inputs (empty selections, incorrect card details) trigger appropriate error messages for correction.
10. **UI Responsiveness:** The layout adapts to mobile, tablet, and desktop screens, ensuring a smooth user experience.
11. **Performance:** Page loads and transitions occur within 2 seconds for a seamless experience.

5.1 Output Screens





Thank you for your booking!

Your event has been successfully booked with **Eventify**. 🎉

ID Booking ID: EVT12345

[Go to Home](#)

6.CONCLUSION

Eventify addresses these challenges by offering a comprehensive, web-based platform that simplifies event planning and management. With its centralized dashboard, dynamic cost calculator, and customizable features, Eventify enables users to plan and manage events efficiently, ensuring they stay within budget and meet their preferences. The intuitive, responsive interface allows event planners to easily navigate the process on both desktop and mobile devices, providing a seamless experience from start to finish. By centralizing all event details, Eventify streamlines the planning process, reduces errors, and enhances the overall user experience, making it the ideal solution for event management.

REFERENCES

To develop the Eventify website, the following technologies and resources were used:

- HTML (Hyper Text Markup Language) – Used for structuring the web pages and defining the layout of content on the home page.
- CSS (Cascading Style Sheets) – Applied for styling the website, including colours, fonts, layouts, and responsive design.
- JavaScript – Implemented for adding interactivity, such as dynamic cost calculations, real-time updates, and smooth navigation.
- Class Notes – Concepts and best practices from class notes were incorporated to enhance functionality, improve user experience, and ensure clean, maintainable code.

<https://github.com/Swathi4224/welcometoevent.git>

