

A Midterm Progress Report

On

Eventify

Submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

COMPUTER SCIENCE AND ENGINEERING

SUBMITTED BY

SHASHANK

SUMEHAR GILL

TARANJIT KAUR

URN – 2104185

URN – 2104200

URN – 2104205

CRN – 2115135

CRN – 2115143

CRN – 2115148

UNDER THE GUIDANCE OF

DR. SITA RANI

Er. GOLDENDEEP KAUR

(October-2024)



Department of Computer Science and Engineering

GURU NANAK DEV ENGINEERING COLLEGE, LUDHIANA

INDEX

| Content | Page No. |
|---|-----------------|
| Chapter 1: INTRODUCTION | 5 |
| 1.1 Event Management System | |
| 1.2 Problems associated with Traditional Event Management Systems | |
| 1.3 MERN stack | |
| 1.4 The Project: Eventify | |
| 1.5 Objectives | |
| 1.6 The following steps are planned to accomplish the project goals | |
| Chapter 2: SYSTEM REQUIREMENTS | 9 |
| 2.1 Software Requirements | |
| 2.2 Software Requirements | |
| Chapter 3: SOFTWARE REQUIREMENT ANALYSIS | 14 |
| 3.1 Problem Statement | |
| 3.2 Modules and Functionalities | |
| Chapter 4: SOFTWARE DESIGN | 17 |
| Chapter 5: TESTING MODULE | 22 |
| Chapter 6: PERFORMANCE OF THE PROJECT DEVELOPED | 25 |
| 6.1 Current Progress and Achievements | |
| 6.2 In-Progress Features | |
| Chapter 7: OUTPUT SCREENS | 27 |
| Chapter 8: REFERENCES | 30 |

LIST OF FIGURES

4.1 Workflow diagram

4.2 User diagram

4.3 zero level DFD

4.4 ER diagram

5.1 Required field for name

5.2 Required email format

5.3 Password must be 8 char long

5.4 Valid username and password for login

7.1 Main window

7.2 Sign up/ Login

7.3 Categories of Events arranged

7.4 About Eventify

7.5 Contact Us

CHAPTER 1

INTRODUCTION

1.1 Event Management System

An Event Management System (EMS) is a software platform designed to streamline the entire process of planning, organizing, executing, and analyzing events. By integrating various digital tools, EMS allows organizers to efficiently manage every stage of an event, from the initial setup to post-event analysis. Key functionalities include event planning and scheduling, which enables organizers to manage multiple events, create detailed agendas, assign venues, and handle timelines.

In addition, EMS handles attendee management by maintaining a centralized database of participant details, including ticket information and personal preferences. The system also facilitates smooth communication between organizers and attendees by sending automated updates, reminders, and notifications, ensuring that attendees stay informed about event details, changes, or important announcements.

Marketing and promotion are another integral part of EMS, allowing organizers to promote events through digital channels such as email campaigns and social media. The system includes analytics to monitor the success of these marketing efforts. EMS also supports venue and resource management, simplifying the allocation of spaces and resources needed for events.

EMS platforms are scalable, making them suitable for various types of events, from small meetings to large conferences, whether held in-person, virtually, or in a hybrid format. By automating several event-related processes, EMS enhances efficiency, improves attendee experiences, reduces operational costs, and provides valuable data for improving future events. Overall, an Event Management System simplifies the complexities of event management, making the process more effective and data-driven.

1.2 Problems associated with Traditional Event Management Systems

Traditional event management systems, often relying on manual processes or disconnected digital tools, present several challenges that make organizing and attending events inefficient.

Below are the primary issues:

Lack of Centralization: Traditional event management systems usually involve multiple platforms or manual processes to handle various tasks such as attendee registration, ticketing, promotions, and post-event feedback. This fragmentation leads to inefficiencies, as event organizers must manage and integrate data from various sources, often resulting in confusion and time loss.

Limited Reach and Promotion: Traditional event management often relies on word-of-mouth or physical advertising (posters, flyers), limiting the reach of promotions. With fewer digital marketing tools integrated into traditional systems, it becomes difficult to engage a larger audience or track the success of marketing efforts.

Inconsistent Communication: Communication between event organizers and attendees in traditional systems is often disorganized. Email threads, phone calls, or paper invitations may result in missed messages, leading to miscommunication about important event updates, such as schedule changes, cancellations, or venue details.

Time-Consuming Registration Process: Registration in traditional systems often involves filling out paper forms or separate online forms that aren't directly linked to the event management system. This process is time-consuming for both attendees and organizers, causing delays in event preparations

1.3 MERN Stack

The MERN Stack is a powerful combination of technologies used to develop full-stack web applications. Each component of the stack is designed to provide a seamless experience in building dynamic web applications. Here is a breakdown of each technology:

MongoDB: MongoDB is a NoSQL database that stores data in a flexible, JSON-like format called BSON (Binary JSON). This structure allows developers to easily store and retrieve large amounts of unstructured data, making it perfect for handling the dynamic data needs of web applications like carpooling systems. MongoDB is also scalable, meaning it can handle increasing amounts of data as the application grows.

Express.js: Express.js is a lightweight web framework for Node.js, used to build server-side applications. It simplifies the process of building APIs (Application Programming Interfaces) that interact with databases and manage HTTP requests. Express.js enables developers to build fast and scalable server-side applications, ensuring efficient communication between the front end and the database.

React.js: React.js is a JavaScript library focused on building user interfaces. It is known for its component-based architecture, allowing developers to create reusable UI components, which helps maintain consistency and improve code maintainability. React offers features like virtual DOM (Document Object Model) that enhance the performance of web applications by updating only the parts of the UI that change, leading to a more responsive and fluid experience.

Node.js: Node.js is a runtime environment that allows JavaScript to be executed on the server side. It is known for its event-driven, non-blocking I/O model, which makes it highly efficient and scalable. Node.js enables developers to build applications capable of handling a large number of concurrent connections, making it ideal for building real-time web applications.

1.4 The Project: Eventify

Eventify is a dynamic web development platform designed to facilitate interactions between users and event organizers, streamlining the event management process. Built using the MERN stack (MongoDB, Express.js, React.js, Node.js), Eventify offers a comprehensive solution where organizers can create, manage, and promote events, while users can register, explore, and participate in them.

Eventify aims to simplify event organization. It serves as a bridge between organizers and attendees, offering features that enhance user experience and operational efficiency. However, Eventify takes this a step further by providing a more customizable and interactive user interface, ensuring that both small-scale and large-scale events can be seamlessly managed.

Key aspects of Eventify include:

- **User Interaction:** Users can register, log in, and engage with events, similar to Quintana Events. They can view details, register for events, and receive updates.
- **Database Storage:** The backend securely stores user and event data, ensuring a smooth and efficient user experience

1.5 Objectives

The objectives of this Eventify are:

1. To facilitate real-time communication and updates between event organizers and attendees.
2. To enable seamless user registration and secure payment integration for events.
3. To provide detailed analytics and reporting to measure event success and to enhance attendee networking through dedicated social features.

1.6 The following steps are planned to accomplish the project goals:

1. Design and implement a secure login and registration system using JWT for session handling, ensuring user credentials are stored securely with hashed passwords.
2. Implement real-time updates.
3. Set up event browsing and selection functionality for users, allowing them to choose events by date, type, and package. Integrate contact options to connect users with event organizers via WhatsApp, Instagram, or phone.
4. Develop a back-end to store the data for the login credentials and the information of the User and their events.
5. Build a secure event booking process with integrated payment gateways such as Stripe or PayPal. Ensure secure payment transactions.
6. Develop a contact us system for users post-event, allowing them to submit reviews and ratings. Implement a review moderation system to ensure appropriate feedback is displayed, and allow organizers to respond to reviews.

CHAPTER 2

SYSTEM REQUIREMENTS

2.1 Software Requirements

The software requirements for the Eventify project include several key development tools, frameworks, and libraries that will form the core of the system's architecture. These technologies are carefully chosen to support the development of a modern, scalable, and high-performance web application.

Integrated Development Environment (IDE): **Visual Studio Code**, a widely used IDE that offers support for JavaScript, React, Node.js, and MongoDB development. With features like syntax highlighting, debugging, Git integration, and customizable extensions, it greatly simplifies development. It is also compatible with extensions for linting, auto-complete, and real-time debugging of MERN stack applications.

Version Control System: **Git & GitHub**, a distributed version control system used for tracking changes in the source code during development. Git allows multiple developers to collaborate on the project efficiently by providing features like branching, merging, and rollback, which enable concurrent development without conflicts. GitHub supports team collaboration with features like pull requests, issue tracking, and code reviews, making it easy to maintain different versions of the project and deploy updates.

Frameworks and Libraries

MongoDB:

- MongoDB is a NoSQL database that stores data in a flexible, JSON-like format called BSON. Unlike traditional relational databases, MongoDB allows developers to store and

retrieve large volumes of unstructured data more efficiently, making it ideal for dynamic applications.

- **Document-Based Model:** MongoDB organizes data into collections and documents. Each document can store various types of data, such as user profiles, ride details, and transaction records, making the database highly adaptable to changes in data structure over time.

Express.js:

- Express.js is a lightweight, flexible web framework built on top of Node.js that simplifies server-side development. Express handles HTTP requests, route management, and middleware integration, allowing developers to quickly build RESTful APIs that connect the front-end with the back-end.
- **Middleware Support:** Express enables the use of middleware functions that execute at different stages of an HTTP request. This is useful for tasks like validating user input, handling authentication, and managing sessions within project.

React.js:

- React.js is a popular front-end JavaScript library used to build user interfaces. React uses a component-based architecture, which allows developers to create reusable UI elements. This modular approach leads to a more organized and maintainable codebase.
- **Virtual DOM:** One of the key features of React is the Virtual DOM, which improves performance by updating only the parts of the web page that have changed, rather than re-rendering the entire UI. This results in faster load times and a more responsive user experience.
- **React Router:** React Router is a library used within React to handle navigation between different pages or components in a single-page application. This is critical, as users need

to navigate between the home page, booking pages, user profiles, and more without refreshing the entire page.

Node.js:

- Node.js is a JavaScript runtime environment that allows developers to execute JavaScript code on the server side. It uses an event-driven, non-blocking I/O model, making it highly efficient and scalable, which is ideal for real-time applications.
- Package Manager (NPM): Node.js comes with NPM (Node Package Manager), which allows developers to easily install and manage third-party libraries and modules. This accelerates development by providing ready-to-use packages for common tasks such as authentication, encryption, and data validation.

Payment Gateway API

We will integrate a secure payment gateway API to handle online payments efficiently. The payment gateway ensures encrypted transactions, providing a secure and trustworthy platform for processing user payments for bookings. It supports multiple payment methods, including credit/debit cards, net banking, and UPI (Unified Payments Interface), offering users a convenient and flexible payment experience.

2.2 Hardware Requirements

The hardware requirements focus on ensuring the availability of robust systems for development and database management. These include development machines for the engineering team and the database infrastructure necessary for storing large amounts of user and ride data.

Laptops/Desktops: Each developer requires a powerful development machine capable of running all necessary software efficiently. The recommended specifications are:

- **Processor:** Intel i5 or higher (or equivalent AMD processor) to handle multitasking and compiling the project efficiently.
- **RAM:** 8 GB (16 GB preferred) to ensure that large applications like Visual Studio Code, MongoDB, and Node.js run smoothly, especially when dealing with large datasets or simultaneous development tasks.
- **Storage:** 500 GB SSD to enable fast read/write operations, especially during database management, deployment, and version control tasks. SSD storage improves the speed of loading large files and libraries, making development more efficient.

Database Server: MongoDB Compass is a graphical user interface (GUI) for MongoDB that allows developers to interact with their data in a visual and intuitive manner. It simplifies data management tasks, providing features like real-time performance monitoring, query optimization, and data visualization, without the need for manual server management. By using MongoDB Compass, developers can efficiently focus on building and maintaining the platform while ensuring that data operations run smoothly.

Features of MongoDB Compass:

- **Data Visualization:** MongoDB Compass offers a powerful visualization tool, enabling developers to view, analyze, and understand the structure of their data, making it easier to query and manipulate collections.
- **Data Security:** MongoDB Compass integrates seamlessly with MongoDB's built-in security features such as encryption at rest and role-based access control. This ensures that sensitive data, including user profiles and event details, remains protected.
- **Query Optimization:** MongoDB Compass allows developers to create, test, and optimize queries in real-time, providing insights into query performance and helping to improve database efficiency.

- **Schema Exploration:** With MongoDB Compass, developers can explore and modify the database schema dynamically, ensuring that the platform's data model evolves with user needs.
- **Performance Monitoring:** MongoDB Compass provides tools to monitor key metrics like query execution time and data usage, helping to identify and resolve performance issues before they affect the platform.

MongoDB Compass serves as an essential tool for managing the database efficiently, ensuring high performance, scalability, and security in the Eventify platform.

CHAPTER 3

SOFTWARE REQUIREMENT ANALYSIS

3.1 Problem Statement

In today's fast-paced world, organizing events, managing attendees, and ensuring seamless communication between event organizers and participants can be highly inefficient using traditional methods. Traditional event management systems often suffer from several issues, including manual data handling, limited reach, fragmented communication, and lack of real-time updates, making it challenging to efficiently plan and execute events.

The problem Eventify aims to solve is the complexity of managing events using outdated systems. There is a need for a centralized, automated platform where event organizers can create, manage, and promote events while attendees can easily register, explore event details, and participate. The platform should streamline various event management processes, from registration and scheduling to real-time communication and feedback collection, while providing real-time analytics and insights to organizers. Eventify's goal is to improve the efficiency and experience of both event organizers and attendees by using modern web technologies, thereby reducing operational costs, minimizing errors, and enhancing engagement.

3.2 Modules and Functionalities

The Eventify platform is built using the MERN stack and consists of several modules that streamline the process of event management. Each module has its own set of functionalities to enhance user and organizer interactions. Below are the core modules and their respective functionalities:

3.2.1 User Authentication and Registration

This module provides secure user registration and authentication features, ensuring that only registered users can access the platform's functionalities.

Functionalities:

- **User Registration:** New users can register by providing their email/username and password.
- **User Login:** Registered users can securely log in using their credentials.
- **Session Management:** Ensures proper handling of user sessions after login.
- **Data Security:** All user data, including login credentials, is securely stored in the database.

3.2.2 Event Registration (For Users)

This module allows users (attendees) to browse, register, and manage their participation in events created by organizers.

Functionalities:

- **Browse Events:** Users can view all upcoming events, including event details like date, time, location, and description.
- **Event Registration:** Users can register for events directly through the platform and receive confirmation via email.
- **View Registered Events:** Users can access a list of the events they have registered for and see any updates or changes.

3.2.3 Notifications and Communication

This module handles all communication between the platform and users, ensuring that attendees and organizers are kept informed about important updates.

Functionalities:

- **Automated Notifications:** The system sends automated notifications to users, including registration confirmations and event reminders.
- **Event Updates:** Users are notified of any changes to the event schedule or details (e.g., venue changes, time adjustments).
- **Real-Time Communication:** Organizers can send real-time messages or updates to registered attendees.

3.2.4 Feedback and Surveys

This module allows organizers to collect post-event feedback from attendees, which can be used to improve future events.

Functionalities:

- **Feedback Collection:** Attendees can provide feedback through custom surveys created by organizers.
- **Survey Customization:** Organizers can tailor feedback forms specific to each event to gather relevant information.
- **Report on Feedback:** Feedback is analyzed and presented to organizers, offering insights into attendee satisfaction and areas for improvement.

CHAPTER 4

SOFTWARE DESIGN

Overview

The Eventify platform is designed to facilitate seamless event planning by connecting users with event organizers. The system is developed using the MERN stack, with the integration of APIs for real-time communication and payment gateways (e.g., Stripe or PayPal). The workflow involves users registering, managing profiles, selecting events, booking services, making secure payments, and submitting feedback upon event completion.

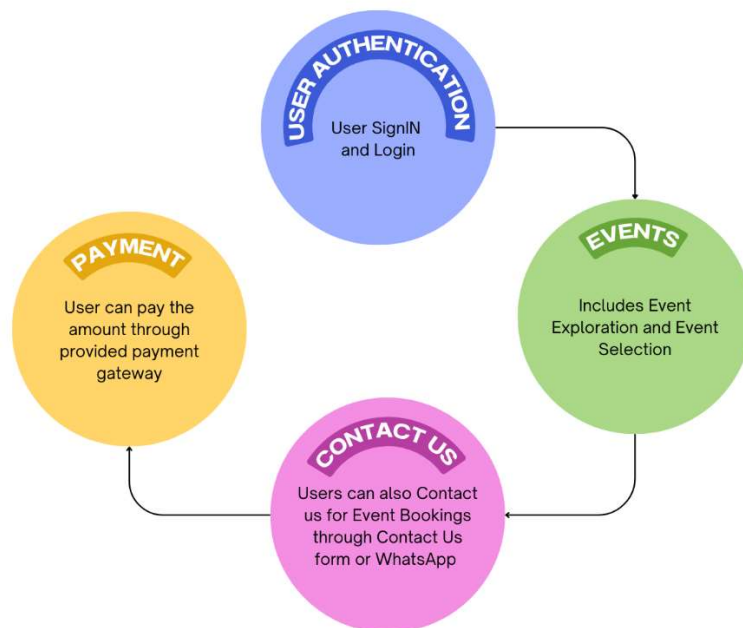


Figure 4.1 Workflow diagram

User Interface

This section outlines the primary functionalities available to users, including the ability to register, log in, manage profiles, browse event services, and book events. Users can also contact

event organizers through integrated communication options like WhatsApp, Instagram, or direct contact numbers. The attached use case diagram visually represents these interactions, demonstrating how users interact with the system to manage their event bookings.

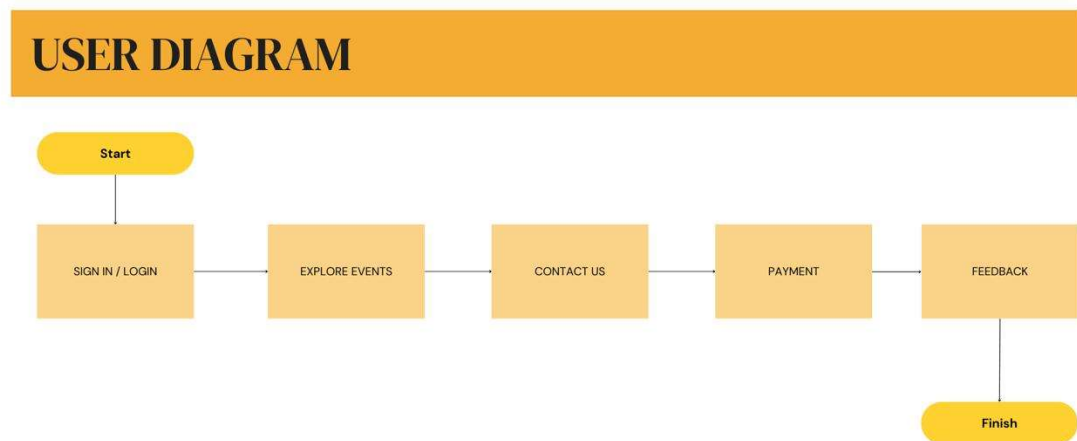


Fig 4.2: User diagram

Data Flow Diagrams (DFDs)

The Data Flow Diagram (DFD) for the Eventify system shows how data moves between users, event organizers, and the admin, as well as various processes within the application. Users can interact with the system by registering or logging in through the User Registration & Authentication process, which grants access to their profiles. Event organizers submit their event details for verification, which are processed by the Admin through the Organizer Event Verification process. Users can then browse, select, and book events through the Event Booking process, with event details stored in the Event Info database. The system also handles transactions through Payment Processing, ensuring secure payments for event bookings.

Level Zero Data Flow Diagram

The following processes are involved in the high-level data flow of Eventify:

- User Registration & Authentication
- User Profile Management
- Contact Us for personalization
- Event Booking & Management
- Payment Processing
- Feedback of Event

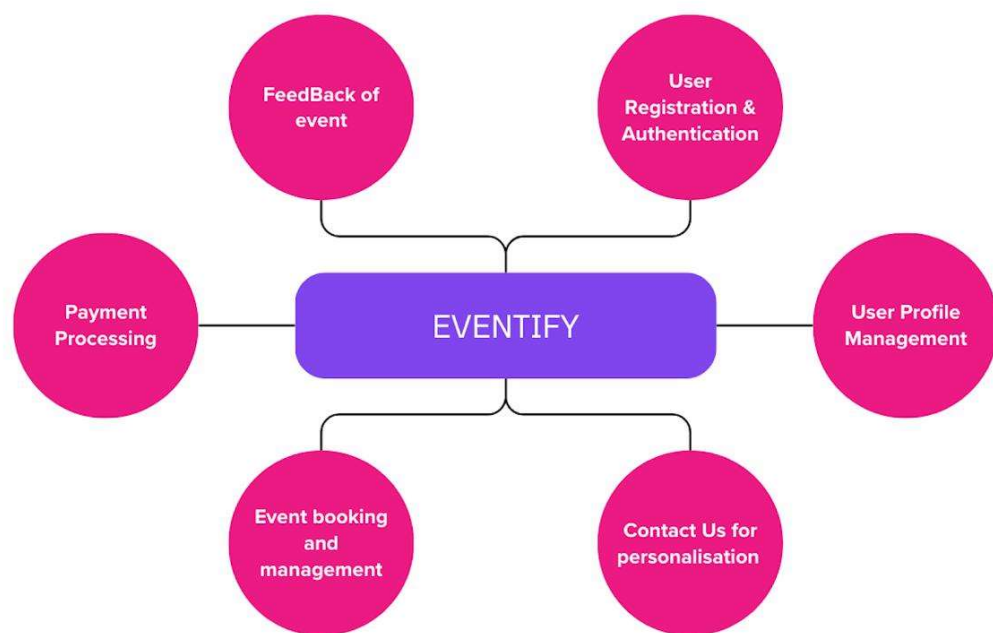


Fig: 4.3 zero level DFD of Eventify

ER Diagram

The ER diagram represents the Eventify system with three main entities: User, Event, and Payment. The User entity contains attributes like personal details, authentication credentials, and roles (organizer or attendee). The Event entity contains event-specific details such as event type, date, location, and packages (Saver, Premium, Luxury). The Payment entity stores payment-related information like the transaction amount, method, and status.

Entities and Attributes:

User:

Primary Key: _id

Attributes: Name, Email, Phone, Password, Profile Photo, etc.

Role: Represents both the organizer and attendee functionalities.

Event:

Primary Key: Event_id

Attributes: Event Name, Date, Location, Packages (Saver, Premium, Luxury).

Role: Stores details about events hosted by organizers.

Payment:

Primary Key: Payment_id

Attributes: Transaction ID, Amount, Payment Method, Status.

Role: Handles payment processing and transaction tracking.

Relationships and Cardinalities:

User-Hosts-Event: A User (1) can host (M) Events.

Cardinality: An organizer can host multiple events, but each event is hosted by one organizer.

User-Books-Event: A User (M) books (M) Events.

Cardinality: A user (attendee) can book multiple events, and an event can be booked by multiple users.

Event-Has-Payment: An Event (1) has (M) Payments.

Cardinality: Each event can have multiple payments, while each payment corresponds to one event.

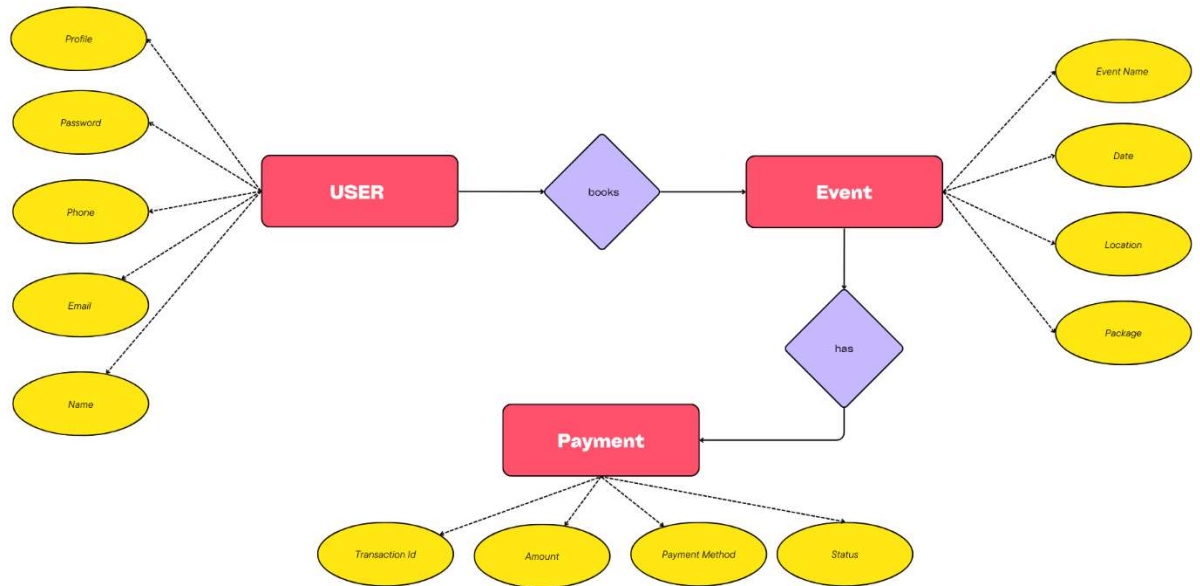


Fig 4.4: ER diagram

CHAPTER 5

TESTING MODULE

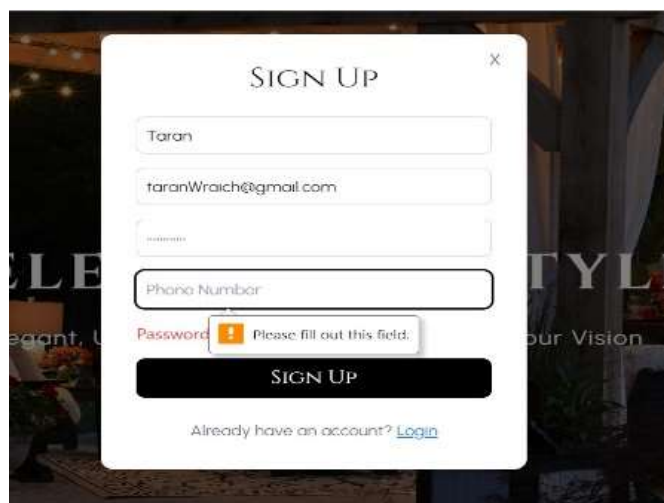
Introduction

The Testing Module is a crucial part of the software development lifecycle for Eventify. It ensures that all functionalities are working as expected and helps identify any bugs or issues before deployment. The testing phase focuses on validating that the system meets the specified requirements and delivers a seamless user experience to both event organizers and attendees.

The primary goals of testing in the Eventify platform include:

- **Functional Testing:** To verify that key modules such as user authentication, event creation, event registration, and notification systems work as expected.
- **Usability Testing:** To ensure that the user interface is intuitive, user-friendly, and that users (both organizers and attendees) can easily navigate through the platform.

Test Case 1: This describes the required field in form which tests for the input field, it shows a message saying “Please fill out this field”.



The image shows a 'SIGN UP' modal form with a close button (X) in the top right corner. The form contains the following fields: a text field for 'Name' with the value 'Taran', an email field with 'taranWraich@gmail.com', a password field, and a 'Phone Number' field. Below the password field, there is a red error message: 'Password [icon] Please fill out this field.' At the bottom of the form is a black 'SIGN UP' button. Below the button, there is a link: 'Already have an account? [Login](#)'. The background of the page is dark with some text visible, including 'Elegant, U', 'TYL', and 'our Vision'.

Figure 5.1 Required field for name

Test Case 2: This describes the email format for the form which shows the required format that is used to fill the details in the form for validation, it shows a message saying “Please include an ‘@’ in the email address.”.

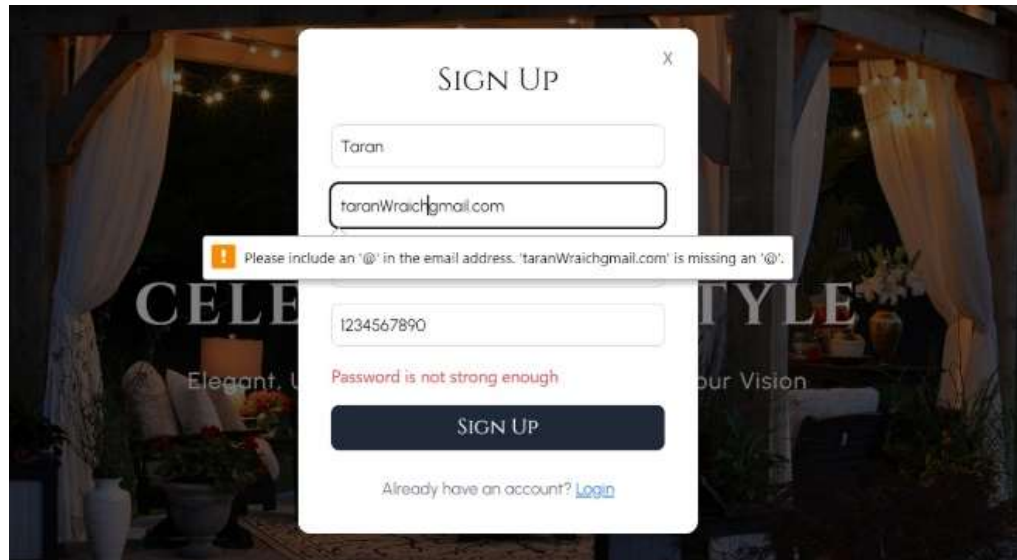


Figure 5.2 Required email format

Test Case 3: This test ensures that the password field enforces a minimum length of 8 characters. If the user enters a password with fewer than 8 characters, it will prompt an error message saying, "Password must contain a minimum of 8 characters."

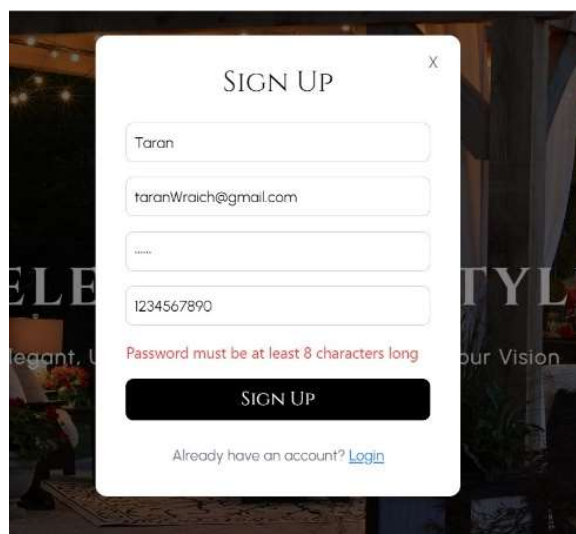


Figure 5.3 password must be at least 8 character long

Test Case 4: This shows invalid login password credentials when a user tries to login with either wrong username or password while login, it shows a message saying “User does not exist”.

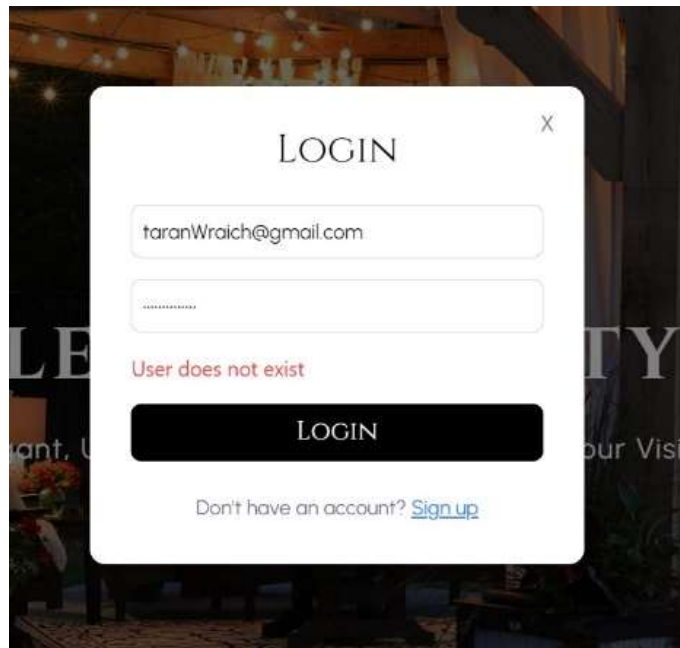


Figure 5.4 Fill the valid username and password for login

CHAPTER 6

PERFORMANCE OF THE PROJECT DEVELOPED

6.1 Current Progress and Achievements

- 1 User Authentication and Profile Management: The platform provides secure user registration and login functionality. User information is securely stored in the database, allowing for future interactions. Users can manage their profiles, with basic details being collected during registration.
- 2 Real-Time Communication Integration: Real-time communication features have been successfully integrated using WhatsApp and Instagram APIs. This enables users and event organizers to interact directly through these platforms, improving responsiveness and ease of communication.
- 3 Database Integration: User data is securely stored in a MongoDB database, ensuring scalability and quick access. The database is designed to handle both user and event information effectively, preparing the system for future expansion.
- 4 Basic User Interface for Event Interaction: A simple and user-friendly interface has been developed, allowing users to register, log in, and navigate through various sections of the platform. The current UI supports interaction with organizers via the integrated communication features and prepares for the future addition of event selection and booking functionalities.

6.2 In-Progress Features

- 1 Event Selection and Booking: Event selection by users, including filtering options by category, date, and location, is still under development. Future improvements will allow users to book events directly through the platform.

- 2 Payment Gateway Integration: Integration of a payment gateway (such as Razorpay or Stripe) is planned to handle event booking transactions. This will ensure seamless and secure payments for users attending paid events.
- 3 User Profile Management & Update: The profile section will be enhanced to allow users to update personal information, upload documents, and manage event preferences.
- 4 Feedback Form Integration: A feedback form system is planned to allow users to provide event feedback and rate the experience. This will contribute to improving the quality and relevance of future events.

CHAPTER 7

OUTPUT SCREENS

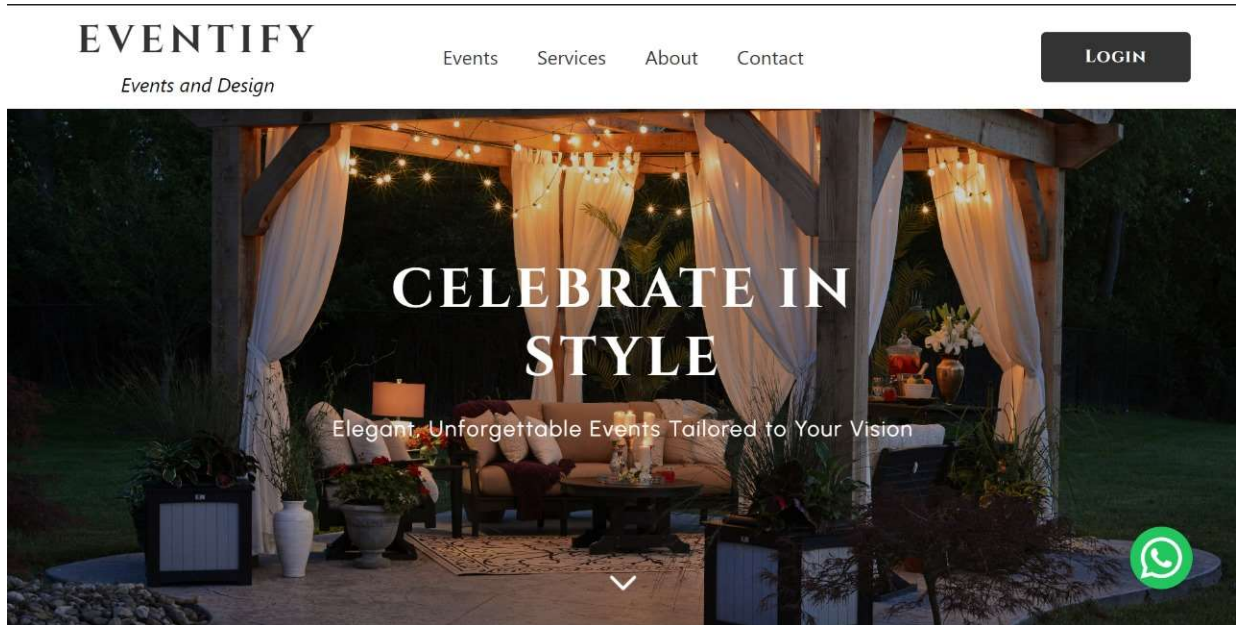


Fig 7.1 Main window

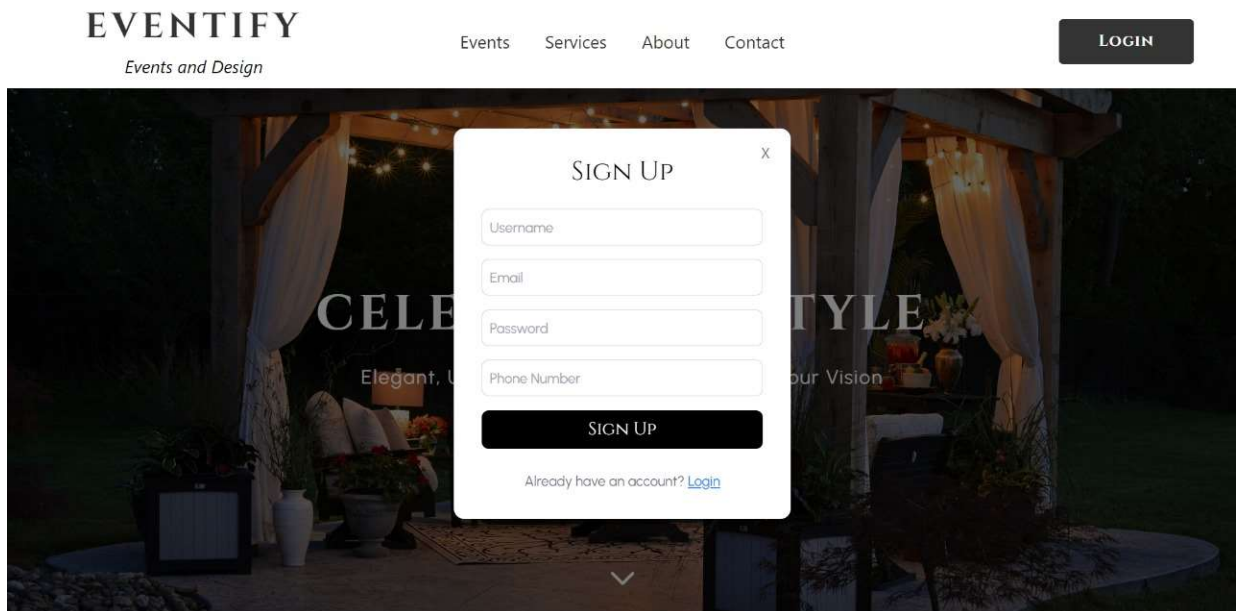


Fig 7.2: Sign up/login window

Event Categories



BIRTHDAY PARTIES

Celebrate your special day with a fun-filled party, customized to your theme and preferences.



ANNIVERSARY CELEBRATIONS

Mark your milestones with elegant, romantic celebrations designed to reflect your love story.



REUNIONS

Reconnect with friends and family in style, with events tailored for nostalgia and fun.



Fig 7.3: Categories of Events arranged by Eventify

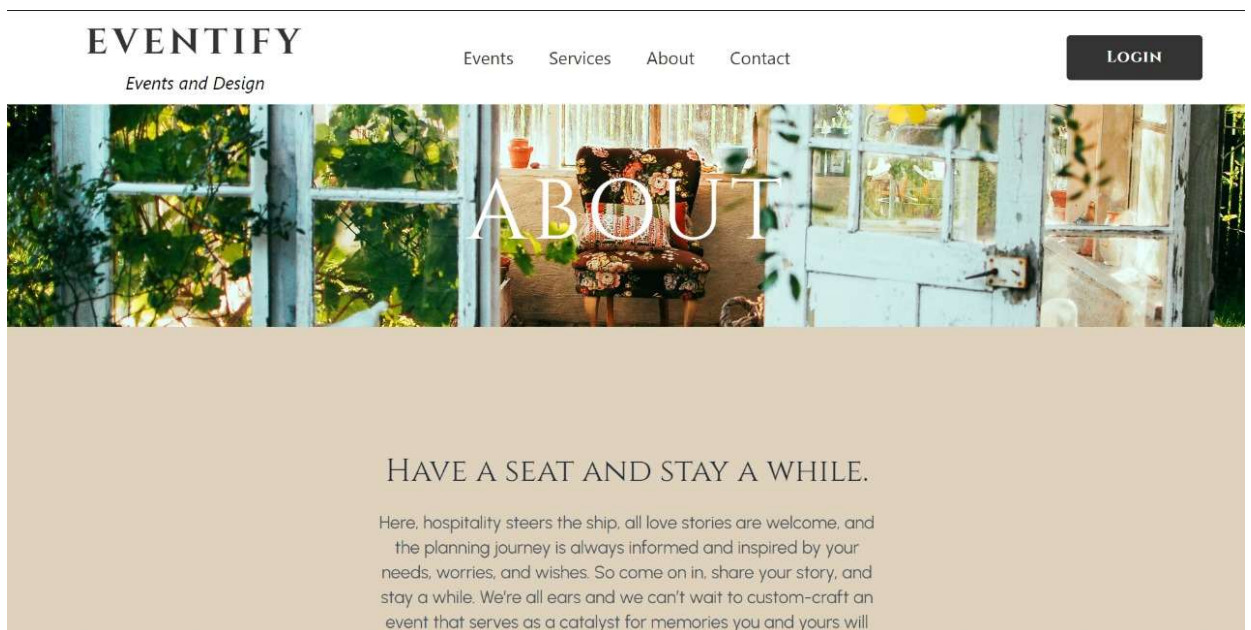


Fig 7.4: About the Eventify

EVENTIFY

Events and Design

Events

Services

About

Contact

LOGIN

CONTACT US

Fill out the form below and we will be in contact shortly.


Name

Email

Phone

Describe Your Event

SEND



123 Event Street, Ludhiana, Punjab

Phone: +91 88476 80989

Email: youreventify@google.com

f

@

t




Fig 7.5 Conatct Us for customization

29

CHAPTER 8

REFERENCES

[1] The Importance of Web Application Architecture: Magnus Andersson

[2] A Systematic Literature Review of the Design Thinking Approach for User Interface

Design: Fardan Zamakhsyari, Agung Fatwanto

[3] Wedding Planner: Poojan Patel, Jishnu Nambiar, Shubham Agarwal and Prof. Neha Kudu.