

EVENTIFY

MAJOR PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD

OF THE DEGREE OF

BACHELOR OF TECHNOLOGY

(Computer Science and Engineering)



Submitted By:

Shashank (2104185)

Sumehar Gill (2104200)

Taranjit Kaur (2104205)

Submitted To:

Dr. SITA RANI

Er. GOLDENDEEP KAUR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GURU NANAK DEV ENGINEERING COLLEGE

LUDHIANA, 141006

November, 2024

Abstract

Eventify is an innovative web-based platform developed to simplify and enhance the process of event discovery, booking, and management. Built on the MERN stack ,Eventify provides a comprehensive solution that connects attendees with event organizers seamlessly. The platform allows users to browse and filter events by category, location, and date, and enables organizers to publish detailed event listings with ease. Secure payment integration ensures smooth and reliable transactions for ticket purchases, while real-time notifications keep users informed of event updates.

With features like user authentication, organizer verification, and a feedback system, Eventify fosters a trusted environment for users and provides valuable insights for organizers through user reviews and ratings. The modular and scalable design allows Eventify to adapt to evolving user needs and accommodate future feature enhancements, such as mobile app support, in-app messaging, and dynamic pricing. Ultimately, Eventify leverages modern web technologies to create a user-friendly platform that enriches the event experience for all participants, streamlining engagement and accessibility within the event industry.

Acknowledgement

We are highly grateful to the Dr. Sehajpal Singh, Principal, Guru Nanak Dev Engineering College (GNDEC), Ludhiana, for providing this opportunity to carry out the major project work.

The constant guidance and encouragement received from Dr. Kiran Jyoti H.O.D. CSE Department, GNDEC Ludhiana has been of great help in carrying out the project work and is acknowledged with reverential thanks.

We would like to express a deep sense of gratitude and thanks profusely to Project Guide Dr. Sita Rani and Er. Goldendeep Kaur, without their wise counsel and able guidance, it would have been impossible to complete the project in this manner.

We express gratitude to other faculty members of computer science and engineering department of GNDEC for their intellectual support throughout the course of this work.

Finally, we are indebted to all whosoever have contributed in this report work.

Shashank

Sumehar Gill

Taranjit Kaur

List of Figures

Figure No.	Figure Description	Page No.
1.1	Introductory Image	2
2.1	Agile SDLC Model	19
2.2	Flow Sequence diagram	21
3.1	Workflow diagram	23
3.2	User Flow Chart	24
3.3	Level 0 DFD	25
3.4	User Event Booking Sequence Diagram	26
3.5	Use Case Diagram	26
3.6	Booking Activity Diagram	27
3.7	User Communication diagram	27
3.8	ER diagram	30
3.9	User Interface Diagram	33
4.1	Required field for name	42
4.2	Required field for valid email	43
4.3	Password must be at least 8 character long	43

4.4	Invalid login credentials	44
4.5	Payment processing with invalid card credentials	44
4.6	Required field during payment	45
4.7	Review submission form	46
5.1	User Interface	48
5.2	User Sign up	49
5.3	User Login	50
5.4	Event categories	50
5.5	Our venues	51
5.6	Adding Review Interface	51
5.7	Review section	52
5.8	Payment Processing Interface	53
5.9	Payment Successful	53
5.10	Transaction history	54
5.11	User Collection Database	55
5.12	Review section – testimonial database	55
5.13	Feedback form database	56

List of Tables

Table No.	Table Description	Page No.
4.1	Stripe API Integration Steps	38

Table of Contents

Contents	Page No.
Abstract	i
Acknowledgement	ii
List of Figures	iii
List of Tables	V
Table of Contents	vi
Chapter 1: Introduction	1
1.1 Introduction to Project	1
1.2 Project Category	2
1.3 Problem Formulation	3
1.4 Identification/Recognition of Need	5
1.5 Existing System	6
1.6 Objectives	8
1.7 Proposed System	8
1.8 Unique Features of the Proposed System	9
Chapter 2: Requirement Analysis and System Specification	11
2.1 Feasibility Study	11
2.2 Software Requirement Specification (SRS) Document	12
2.3 SDLC Model to be Used	17
Chapter 3: System Design	22

3.1 Design Approach	22
3.2 Detailed Design	22
3.3 User Interface Design	31
3.4 Methodology	33
Chapter 4: Implementation and Testing	36
4.1 Introduction to Languages, IDEs, Tools, and Technologies Used	36
4.2 Algorithm/Pseudocode Used	38
4.3 Testing Techniques	40
4.4 Test Cases Designed for the Project	42
Chapter 5: Results and Discussions	47
5.1 User Interface Representation	47
5.2 Snapshots of System with Brief Detail of Each and Discussion	48
5.3 Back-End Representation (Database)	54
Chapter 6: Conclusion and Future Scope	57
References	59
Appendix A: Development Environment	60PP

Chapter 1: Introduction

1.1 Introduction to Project

The Eventify project was a significant endeavour to develop a cutting-edge online booking and event management platform. The major objective was to create a smooth, effective, and intuitive system that makes it simple for users to communicate with event planners and offers a streamlined booking and event management experience. This software is made to serve a broad range of users, including people and businesses seeking a dependable platform to handle different kinds of events without the hassles of conventional event planning.

Eventify was created with the MERN stack, which consists of MongoDB, Express, React, and Node.js. It makes use of these technologies to offer a high-performance, scalable application. Through integrated Instagram and WhatsApp contact options, the site offers real-time communication capabilities that let users get in touch with organisers directly for prompt updates and inquiries. Additionally, Eventify features secure user registration and login, with data securely stored in the backend database to maintain user information and event preferences.

Eventify's interface is both intuitive and rich in functionality, ensuring that users can effortlessly navigate the platform. Users can register, log in, and view available events, select options, and book services through the secure payment system, which integrates reliable payment gateways like Stripe or PayPal. Furthermore, the platform includes a feedback and review feature, allowing users to share their event experiences and provide valuable insights to improve future services.

In response to the demand for streamlined event management solutions, Eventify serves as a versatile tool for organizing and managing events, solving issues such as fragmented planning, time-consuming communications, and limited customization options. By offering a centralized platform for event booking and management, Eventify aims to simplify the event planning

process for users and organizers alike, enhancing efficiency and delivering a user-centered experience.

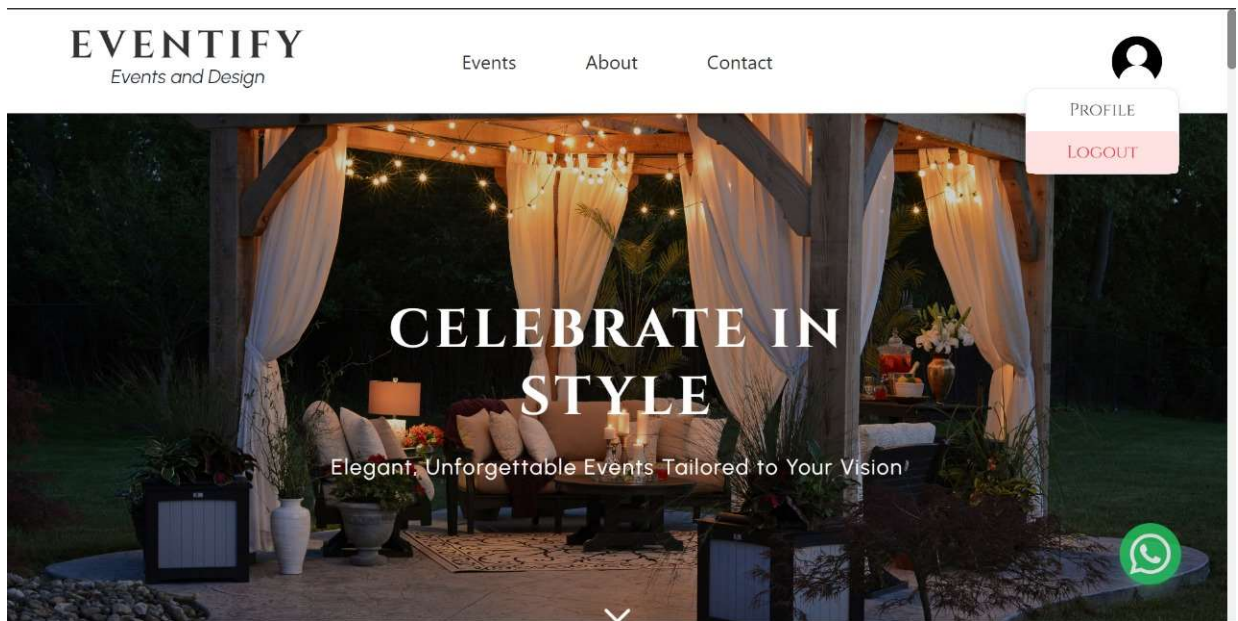


Figure 1.1 Introductory Image

1.2 Project Category

Classified as an application-based project, the Eventify project is developed as a fully working online tool for event administration. Its function as a useful, interactive tool that helps people and organisations expedite event planning and booking is highlighted by this classification. Designed for anyone who want to plan or attend events, Eventify offers a user-friendly, easily navigable platform for organising event information, registration, and feedback.

By providing a single platform for linking users and event organisers, Eventify aimed to streamline the booking and management process for events. Similar to well-known booking systems, it offers users the ease of online event management with features like safe payments and real-time communication that are specifically designed for a variety of event requirements. The project was designed to meet several essential application-oriented requirements:

- Technical Integration for Practical Use: The platform integrates secure registration and login functionalities to maintain user privacy and data integrity. Contact buttons for WhatsApp and Instagram facilitate real-time communication with event organizers, enhancing user interaction. Furthermore, Eventify incorporates a secure payment gateway for seamless transaction processing, ensuring that users can conveniently make payments within the platform when booking events.
- User Interface and User Experience (UI/UX): The user interface was crafted to deliver an intuitive and engaging experience for general audiences. The interface includes features such as user registration, profile management, event browsing, booking, and feedback submission. These features were designed to make it easy for users to register, select events, and engage with event organizers directly from the platform.
- Security and Credibility: Recognizing the importance of security and trust in event management, Eventify implements user authentication and secure data storage, protecting user information and transactions. The platform also offers feedback and review capabilities, allowing users to evaluate events and build community trust. These features collectively ensure a reliable and secure environment, enhancing credibility among users.

1.3 Problem Formulation

The growing need for simplified, easily accessible, and effective event management solutions led to the creation of the Eventify project. Conventional systems for event booking and organising frequently have issues that make the process cumbersome, disjointed, and unappealing to contemporary consumers. An extensive description of each problem element and how these difficulties were included into Eventify's problem formulation can be found below:

Communication That Is Broken: Communication between planners and guests is frequently fragmented in traditional event management, depending on many platforms that impede prompt response times and efficient exchanges. Confusion and a decline in attendance satisfaction might

result from misunderstandings and delays in event updates. In order to solve this problem, Eventify incorporates real-time communication capabilities with buttons for Instagram and WhatsApp, enabling people to get in touch with event organizers instantly, fostering a more connected and responsive user experience.

Complex Registration and Payment Processes: Conventional event registration systems can be labor-intensive and may require users to navigate through multiple forms or platforms to secure a booking, which reduces ease of use. Payment processing often lacks security and convenience, creating an additional barrier for users. Eventify simplifies the registration process with a single platform where users can seamlessly register for events, while also integrating a secure payment gateway to enable efficient and safe transactions.

Limited User Feedback and Engagement: Traditional event management systems typically do not provide structured ways for attendees to offer feedback or engage with event organizers post-event. This lack of feedback hinders organizers from improving future events based on user experiences. Eventify incorporates a feedback and review system, allowing users to share their experiences, helping organizers gain insights into attendee preferences and improve their offerings.

High Cost and Lack of Customization in Event Packages: Users often encounter hidden or excessive costs in traditional event planning, alongside limited options for customizing event packages to fit their needs. Eventify addresses this by offering transparent pricing and customizable event packages (Saver, Premium, Luxury), making it easier for users to select packages that match their preferences and budget.

Trust and Credibility Concerns: In the event industry, trust and reliability are crucial, especially when booking with unfamiliar organizers or vendors. Users may be uncertain about the quality and reliability of service providers, resulting in hesitancy. Eventify builds trust by curating a list

of verified and trusted vendors, ensuring that users can confidently book events. The platform also includes secure authentication measures to maintain user privacy and data protection.

1.4 Identification/Recognition of Need

In recent years, the shift towards digital solutions for event planning highlighted the lack of accessible, efficient, and user-friendly platforms to streamline the complex process of organizing and booking events. This section outlines the key factors that drove the development of Eventify and the specific needs it was designed to address.

Rising Event Expenses and Hidden Fees: Both event planners and attendees experienced financial distress as a result of high event planning expenses, which included venue reservations, vendor fees, and other charges. These costs frequently prevented people, particularly those with little funds, from organising the events they had in mind. By providing customisable event packages (Saver, Premium, and Luxury) and transparent pricing, Eventify addressed this issue by letting users choose solutions that fit their budgets and minimising hidden costs through up-front and open pricing.

Demand for Simplified Event Planning and Coordination: Conventional event planning frequently necessitates extensive logistical preparation and coordination with numerous vendors, which may be burdensome and ineffective. Eventify responded to this need by integrating vendor services, making it easy for users to plan events by selecting from a curated list of trusted vendors all in one place. This consolidated approach simplified event management, saving users time and effort.

Trust and Security Concerns in Event Bookings: Many traditional event booking platforms lack the verification measures necessary to ensure the credibility of vendors and organizers, leading to concerns about the quality and reliability of services. Eventify includes a system for verifying vendors and organizers, providing users with a trustworthy platform where they can confidently

book events and interact with verified service providers. This emphasis on credibility builds trust and enhances user satisfaction.

Growing Desire for Real-Time Communication and Digital Convenience: In the digital age, users expect a seamless, tech-driven experience for all services, including event planning. Traditional methods of event coordination, often reliant on email or phone communication, are seen as outdated and cumbersome. Eventify caters to this demand with a user-friendly interface and real-time communication tools, including WhatsApp and Instagram buttons that allow attendees to directly contact organizers. Secure online payment integration also ensures a smooth booking process, providing users with the convenience of digital payment options.

Need for Feedback and Community Engagement: Event organizers often lack structured feedback from attendees, limiting their ability to improve future events. Additionally, attendees value the opportunity to share their experiences with the community. Eventify includes a feedback and review system where users can provide valuable insights post-event, helping organizers improve their services and fostering a connected, engaged community.

1.5 Existing System

The existing event management and booking systems before the development of Eventify had notable limitations that reduced their effectiveness and hindered widespread adoption. Here's a look at the limitations of current digital event management applications and the challenges these systems faced in addressing modern organizer and attendee needs.

Communication that was Inconsistent and Fragmented: Conventional event management systems frequently relied on manual communication channels like phone conversations, emails, and unofficial messaging groups. These systems lacked the effectiveness required to successfully link organisers and participants in the absence of a centralised communication platform. Attendees frequently had trouble getting timely updates or pertinent event information, which resulted in a

subpar user experience. For major events that needed coordination and real-time information, this discrepancy proved particularly troublesome.

Limited user interface options and a poor user experience were features of traditional event systems, which frequently required customers to browse between pages or systems for communication, payment, and event registration. It was difficult for participants to register, handle reservations, and get in touch with organisers given the absence of a single platform. The lack of a formal feedback system, user-friendly navigation, or real-time updates resulted in user frustration, missed opportunities, and an overall inconvenient experience.

Absence of Verification and Trust: Traditional event booking mechanisms raised serious questions about safety and trust. Because many unofficial setups lacked any kind of vendor or organiser credibility check, users were left in the dark about the validity or quality of services. Attendee involvement was hampered by this lack of trust since people were reluctant to sign up for events with unreliable organisers. Without mechanisms to confirm government-issued identification, venue certifications, or vendor quality requirements, user and vendor verification remained scanty, even on some digital platforms.

Limited Payment Options: In the past, traditional event booking systems frequently required cash or handwritten payments, which resulted in issues and disagreements over amounts, particularly when dealing with large transactions or events involving multiple vendors. This reliance on cash reduced transparency and added to the risk of errors in fare or service payments. Although some digital event systems began offering online payment options, they often lacked the flexibility, security, or transparency needed for modern users to feel confident about secure transactions.

Minimal Data-Driven Insights: Traditional and even some digital event management systems lacked the ability to provide data-driven insights, such as tracking user engagement, registration history, or attendee preferences. Without data analytics and user feedback mechanisms, these platforms missed opportunities to improve the event experience, measure success metrics, and

enhance future planning through actionable insights. Eventify fills this gap by integrating analytics, enabling organizers to gain valuable insights into attendee engagement and satisfaction, leading to better event customization and success measurement.

1.6 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.

1.7 Proposed System

The proposed Eventify system was designed as an innovative, secure, and user-friendly event management and booking platform, addressing the limitations of traditional event planning systems. Below is a detailed breakdown of Eventify's main features, design, and functionalities.

Real-Time Communication and Updates: One of Eventify's core features was real-time communication between attendees and event organizers. The platform included buttons for WhatsApp and Instagram, allowing attendees to directly contact organizers for immediate updates or inquiries. This streamlined communication minimized delays in obtaining information and ensured users were always up-to-date with event details.

User Registration and Authentication: Eventify provided a secure user registration and authentication process to create a trusted platform for attendees and organizers. Users registered on the platform, and their information was stored securely within the database. This enabled a

streamlined, personalized experience for attendees, with access to account-based features such as saved events, feedback, and past bookings.

Intuitive User Interface (UI): The Eventify platform featured a visually appealing and user-friendly interface, designed to facilitate seamless interaction. The UI was organized into sections that allowed both organizers and attendees to easily navigate the platform:

- **Event Search and Booking:** Attendees could access a search interface to browse events, filter by date, category, or location, and view event details like schedule, venue, and ticket options.
- **Feedback and Review System:** Users were able to provide feedback and reviews on events they attended, allowing future attendees to make informed decisions and organizers to improve their offerings.

Integrated Payment Gateway: A significant limitation in prior event systems was the lack of secure, digital payment options. Eventify included an integrated payment gateway, allowing users to pay for event bookings within the platform using credit/debit cards, UPI, and other digital payment methods. The platform used encryption protocols to protect sensitive information, ensuring all transactions were secure and private.

1.8 Unique Features of the Proposed System

The Eventify platform was designed with several unique features to differentiate it from existing event management systems, addressing user needs for seamless communication, security, usability, and flexibility.

Comprehensive User Verification for Trust and Security: Eventify prioritizes user trust and platform security through a robust verification process. Organizers are required to submit

documentation, which is reviewed and verified by administrators through a dedicated admin interface. This ensures that only credible organizers are allowed to host events on the platform, providing attendees with added confidence in the quality and authenticity of the events.

Real-Time Communication Integration: One of Eventify's standout features is its real-time communication system, allowing attendees to connect with event organizers directly through WhatsApp and Instagram buttons. This feature enables instant communication for quick responses to inquiries, updates, or special requests, enhancing user engagement and satisfaction.

Interactive and User-Friendly Interface: The platform boasts an intuitive, visually appealing interface that is accessible to users of all technical backgrounds. The interface is organized into clear sections for registration, event browsing, booking, and profile management, ensuring easy navigation. Real-time notifications, simplified forms, and streamlined icons allow users to perform actions like booking events, updating profiles, and submitting feedback effortlessly.

Secure and Flexible Payment Gateway Integration: The platform incorporates a secure payment gateway, supporting various digital payment methods, including credit/debit cards, UPI, and e-wallets. This allows users to make payments directly within the platform without relying on external methods. Encryption safeguards user data, and features like transaction tracking ensure smooth and secure financial transactions.

Feedback and Review System for Continuous Improvement: Eventify encourages post-event feedback, allowing attendees to share their experiences and rate events. This feedback is valuable for organizers to improve future events and helps attendees make informed decisions. The review system also includes moderation to ensure that only constructive feedback is displayed, supporting a positive community atmosphere.

Chapter 2: Requirement Analysis and System Specification

2.1 Feasibility Study

The feasibility study for Eventify examines whether the project is achievable and sustainable.

Technical Feasibility

The MERN stack (MongoDB, Express.js, React, and Node.js) is used by Eventify to create a secure, scalable, and reliable web application. User identification, data storage, real-time communication, and event management are all skilfully handled by this full-stack solution.

APIs and Integrations: Users and event planners may communicate easily thanks to integration with the Instagram and WhatsApp APIs. Furthermore, a payment gateway (Stripe) is integrated to facilitate safe and efficient transactions, resulting in seamless and intuitive payment procedures.

Scalability and Compatibility: As the application expands, the platform can support more users and new features thanks to the scalability offered by the MERN stack architecture. Additionally, its cross-platform interoperability lays the groundwork for future growth into mobile applications.

Economic Feasibility

Cost-Effectiveness: Eventify reduces development expenses by utilising open-source technologies like Node.js, MongoDB, and React. Expected costs include server resources, Stripe transaction fees, and hosting, including MongoDB Atlas for the database.

Revenue Generation: Transaction fees for sponsored events, premium memberships for access to special features, and possible advertising relationships with sponsors and vendors are some ways that Eventify can make money.

Long-Term Financial Sustainability: Eventify can maintain long-term economic sustainability by introducing subscription options as the platform's user base expands.

Operational Feasibility

User Training and Interface Design: Eventify's intuitive user interface reduces the need for extensive training, making it easy for new users to navigate. Admin users have access to a simple and intuitive dashboard to manage user verifications, review user activity, and handle feedback.

Maintenance and Updates: The modular structure of Eventify enables seamless updates and maintenance. Using version control systems such as GitHub and the latest development frameworks, the platform can be improved without impacting the user experience.

Customer Support and Documentation: Basic user documentation and FAQs are available to assist users in understanding the platform's functionalities. Additionally, a help desk or support system managed by the admin team can address user concerns as they arise.

2.2 Software Requirement Specification (SRS) Document

The SRS document for Eventify specifies the detailed requirements to ensure that the system meets user expectations and provides a seamless experience.

Data Requirements

User Information: Each user must provide personal details such as name, email, contact number, and other basic profile information for verification. Event organizers must submit additional details about their services and credentials.

Event Data: The system must store event information, including event name, date, location, type, package options (Saver, Premium, Luxury), and organizer details.

Payment Information: Transaction details for each booking, including the amount, payment method, and transaction status, should be securely stored to facilitate secure and reliable payments.

Rating and Review Data: User feedback for each event, including ratings and comments, should be stored to maintain service quality and accountability.

Functional Requirements

User Authentication: The system must allow users to register, log in, and manage their profiles. A secure authentication system (OAuth) should prevent unauthorized access, with additional login options provided.

Event Management: Organizers should have features to publish and manage events, while users can search, view details, and register for events based on date, location, and type.

Admin Document Verification: Admins should have access to a panel for reviewing, approving, or denying user-uploaded documents and managing event listings.

Payment Gateway Integration: A secure, reliable payment gateway (Stripe) to process event payments, providing transaction confirmation to both organizers and users.

Review and Feedback: Users can rate and review events after completion, allowing organizers to monitor user satisfaction and identify improvement areas.

Performance Requirements

Speed: Eventify should load pages and update event information within 2 seconds. API requests should be optimized to ensure efficient real-time data updates.

Scalability: The system must support a growing user base, initially targeting 1,000 concurrent users, with future scalability to support higher traffic.

Availability: Eventify should aim for 99.9% uptime to ensure high availability, particularly during peak event planning and registration periods.

Dependability Requirements

Reliability: Eventify should reliably store and retrieve user and event data, ensuring no data loss, even during high usage.

Fault Tolerance: The system should handle errors gracefully, including payment issues and data retrieval errors, with clear error messages guiding users through corrective actions.

Maintainability Requirements

Modular Codebase: The MERN stack allows for a modular structure, making it easy to implement updates and fixes. Future upgrades and third-party integrations can be done without significant disruption.

Documentation: Each module should be well-documented, with GitHub used for version control to facilitate collaboration and maintenance.

Security Requirements

User Authentication and Authorization: Passwords should be securely hashed, and the system should support OAuth-based login for added security.

Data Encryption: All sensitive user data, including payment details and personal information, should be encrypted during storage and transfer.

Admin Access Control: Access to sensitive functionalities, like document verification and user management, should be restricted to authorized personnel.

Look and Feel Requirements

User Interface (UI): A clean, minimalistic, and intuitive design with clear icons, consistent colors, and readable fonts provides a welcoming experience. The interface should be user-friendly for both organizers and attendees.

Responsiveness and Brand Consistency: The design should be responsive, adapting seamlessly to desktops, tablets, and mobile devices, while reflecting a consistent, professional brand image.

Software Requirements

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

Integrated Development Environment (IDE): Visual Studio Code is recommended for development, offering support for JavaScript, React, Node.js, and MongoDB, with features like syntax highlighting, debugging, and real-time linting, enhancing development efficiency.

Version Control System: Git & GitHub are used for tracking source code changes and team collaboration, offering features like branching, merging, and pull requests to maintain code integrity and deploy updates effectively.

Frameworks and Libraries

MongoDB:

- NoSQL Database: MongoDB stores data in a flexible, JSON-like format (BSON), making it ideal for managing dynamic event and user data.
- Document-Based Model: MongoDB organizes data into collections and documents, storing various types of data, such as user profiles, event details, and transaction records, ensuring adaptability to changing data structures.

Express.js:

- **Web Framework:** Express.js simplifies server-side development, handling HTTP requests, route management, and middleware integration for building RESTful APIs to connect the front end with the back end.
- **Middleware Support:** Express enables middleware for tasks like validating user input, handling authentication, and managing sessions.

React.js:

- **Front-End Library:** React.js is used for building user interfaces with a component-based architecture, creating reusable UI elements that result in a maintainable and organized codebase.
- **Virtual DOM:** React's Virtual DOM optimizes performance by only updating the parts of the page that change, ensuring a responsive user experience.
- **React Router:** Allows smooth navigation between pages or components without page refresh, critical for navigating between the home page, event listings, and user profiles.

Node.js:

- **JavaScript Runtime:** Node.js allows JavaScript code to run on the server side, using a non-blocking I/O model for efficient and scalable real-time applications.
- **NPM (Node Package Manager):** Provides ready-to-use packages for common tasks like authentication, encryption, and data validation, accelerating development.

Stripe API:

- **Payment Gateway:** Stripe is integrated for handling secure online payments, supporting multiple payment methods, including credit/debit cards and digital wallets, providing flexibility and security for transactions.

Hardware Requirements

The hardware requirements ensure robust systems for development and database management, necessary for handling high volumes of event and user data.

Laptop/Desktop: Each developer requires a powerful development machine capable of running all necessary software efficiently. Recommended specifications include:

- **Processor:** Intel i5 or higher (or equivalent AMD processor).
- **RAM:** 8 GB minimum, with 16 GB preferred.
- **Storage:** 500 GB SSD to ensure fast read/write operations, especially during database management and version control tasks.

Database Server: MongoDB C is recommended for managed cloud hosting, providing high performance, scalability, and security without manual configuration.

- **Automated Backups:** Ensures data safety with scheduled backups.
- **Data Security:** MongoDB Compass includes encryption and role-based access control for sensitive user data.

Performance Monitoring: Real-time monitoring and alerting help maintain optimal performance and address any issues promptly.

2.3 SDLC Model to be Used

For the development of Eventify, the Agile SDLC Model was selected due to its adaptability and iterative approach, making it ideal for web-based applications that require ongoing improvements and responsiveness to user feedback.

Why Agile SDLC for Eventify?

Iterative Development: Thanks to Agile, the Eventify platform can be developed in incremental stages, thus fostering the ability to integrate changes due to user needs, input, and/or testing feedbacks efficiently. Owing to this iterative development process, it is expected that every feature developed is for the current user demands.

User-Centric Approach: In an Agile development, user feedback is solicited and incorporated throughout the development cycle meaning every Eventify module, including event search and booking to payments, exceeds the satisfaction of users and positively adds value to the experience.

Flexibility and Adaptability. Agile in particular allows for the incorporation of new features and changing the GUI, as well resolving new emerging requirements which include extra means of communication or an update of the payment system without stymying the progress of the ongoing project.

Quality Assurance and Testing: Quality issues in projects developed in the Agile manner are not afterthought as each project sprint integrates a testing cycle which helps detect and fix bugs and performance problems very early.

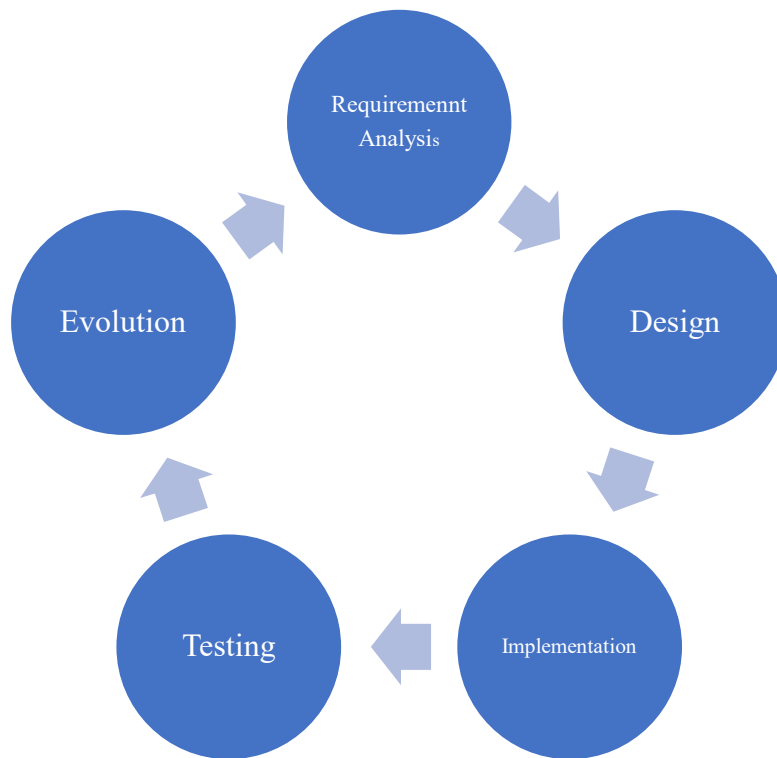


Figure 2.1 Agile SDLC Model

Agile Phases in the Eventify Project

Requirement Gathering and Planning - The process begins with an in-depth study of existing systems and technology analysis to gather requirements and understand the scope of the project. This phase helps in identifying key features needed for Eventify, such as real-time communication, event booking, and payment integration.

Design and Development - The UI and database schema are designed to establish the structural and visual aspects of the platform. Development follows, starting with the implementation of the user interface. Each component, including user authentication, real-time communication, event booking features, and review functionality, is developed and tested iteratively to ensure functionality and compatibility.

Testing and Quality Assurance - During each phase, rigorous testing is conducted to ensure all features work as expected. Functional tests verify individual features like booking and payment, while integration tests confirm the smooth interaction between modules.

Deployment and Review - After completing each feature, the Eventify platform is deployed to a staging environment for user feedback and testing. This allows the team to gather insights on user experience and make necessary adjustments.

Maintenance and Iteration - Based on user feedback and performance insights, optimizations are made in subsequent iterations. This iterative process continues until the project's goals are fully



Figure 2.2 Flow Sequence diagram

Chapter 3: System Design

3.1 Design Approach

The design approach for Eventify follows an Object-Oriented Design (OOD) methodology. This approach is suitable as it allows encapsulating functionalities into distinct modules, which aligns with the modular architecture of the platform. The system is divided into several interconnected objects, such as User, Event, Payment, and Admin, each encapsulating data and operations related to specific functionalities.

Advantages of Object-Oriented Approach:

Modularity: Each thing, inclusive of User or Event, may be evolved and maintained independently, helping Agile development. This modularity enhances flexibility, permitting person modules to be stepped forward or accelerated without affecting the general gadget.

Reusability: Components may be reused or extended for future functionalities, including including new event categories, loyalty applications, or extra payment strategies. This reusability minimizes code duplication and improves maintainability.

Scalability: As Eventify grows, OOD facilitates the addition of recent features, along with a mobile app, advanced occasion suggestions, or greater communication alternatives, making it less difficult to scale the platform to satisfy growing user needs and various requirements.

3.2 Detailed Design

This section provides detailed design documentation for the project using structured analysis and various diagrams. The design addresses each of the project's objectives through detailed illustrations.

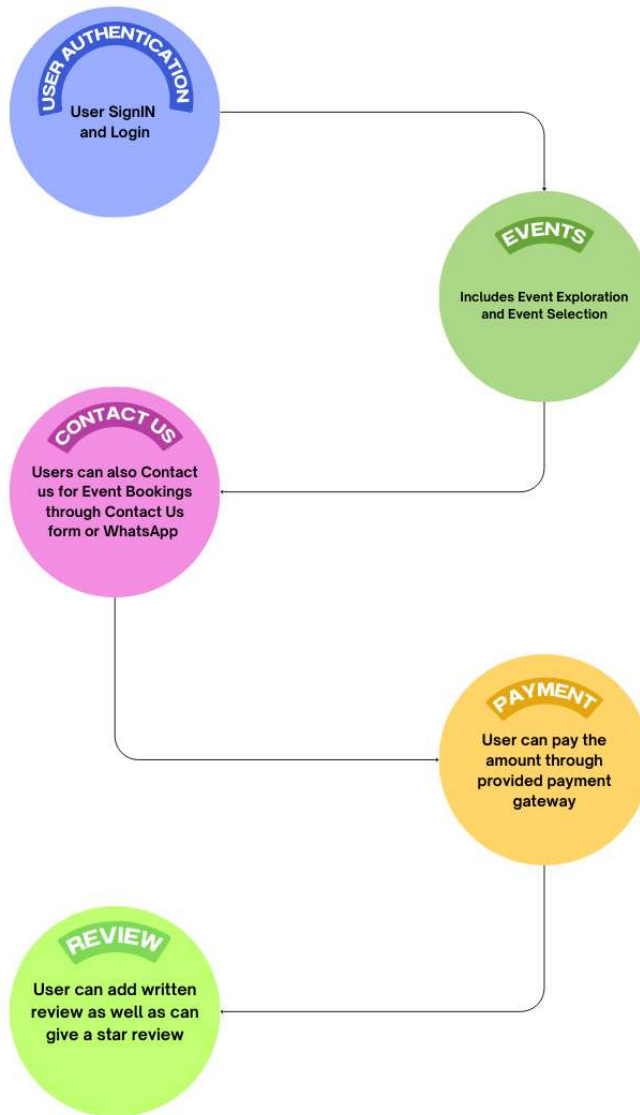


Figure3.1 Workflow diagram

3.2.1 Flow Charts / Block Diagrams

Purpose: Outlines the comprehensive flow from user registration, event browsing, and selection, to event booking and payment, as well as the administrative flow for event management and review verification.

Description: Users start by signing in or registering on Eventify. Once authenticated, they can browse available events, select specific events to participate in, and provide ratings and reviews

after event completion. For event booking, users proceed through a secure payment process, integrated with Stripe, ensuring a seamless experience.

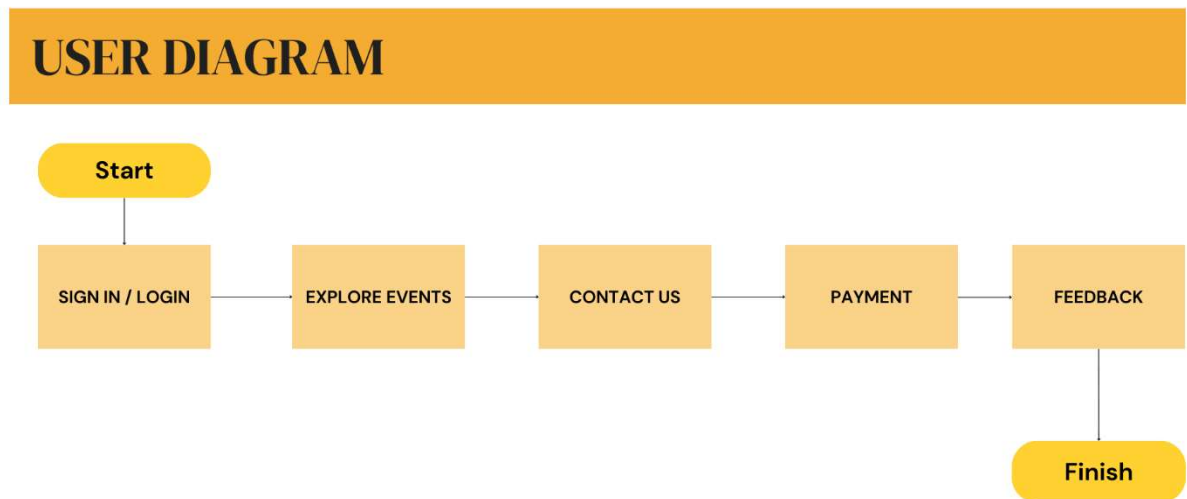


Figure 3.2 User Flow Chart

3.2.3 DFDs (Data Flow Diagrams)

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 0 DFD

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.



Figure 3.3 Level 0 DFD

3.2.4 Sequence Diagrams

User Event Booking Sequence

Participants: User, Eventify System, Event Database, Payment Gateway.

Flow: The user initiates a search, the system retrieves events, confirms booking, and completes payment.



Figure 3.4: User Event Booking Sequence Diagram

3.2.5 Use Case Diagram

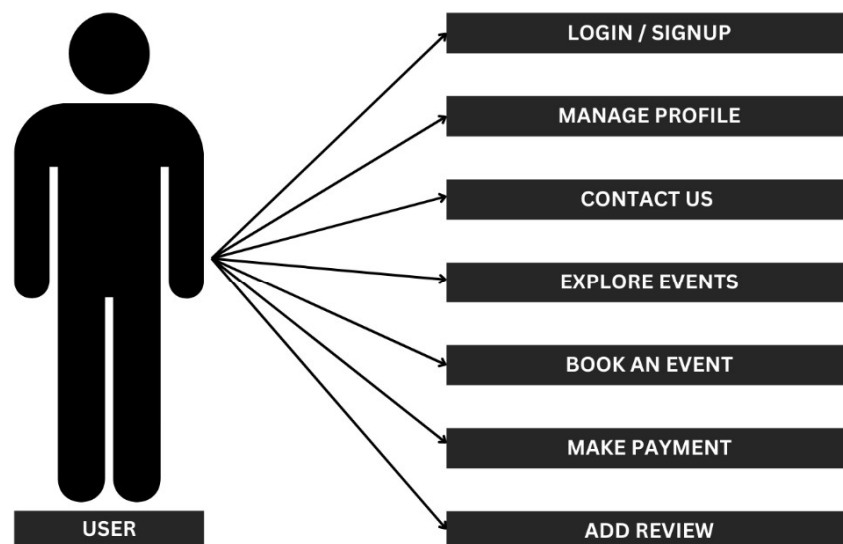


Figure 3.5 Use Case Diagram

3.2.6 Activity Diagrams

Booking Activity Diagram



Figure 3.6: Booking Activity Diagram

3.2.7 Communication Diagram

User and Event Booking Communication

Interactions: Details interactions between User, System, and Database.

Flow: The user request is relayed through each component to retrieve, display, and confirm event bookings.

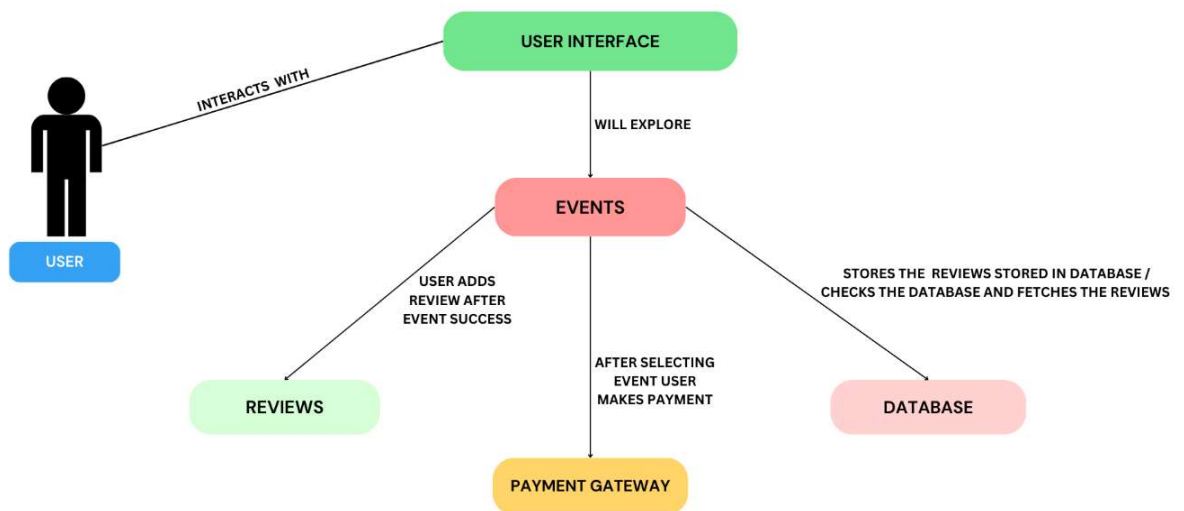


Figure 3.7: User Communication Diagram

3.2.8 Database Design: 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. 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.

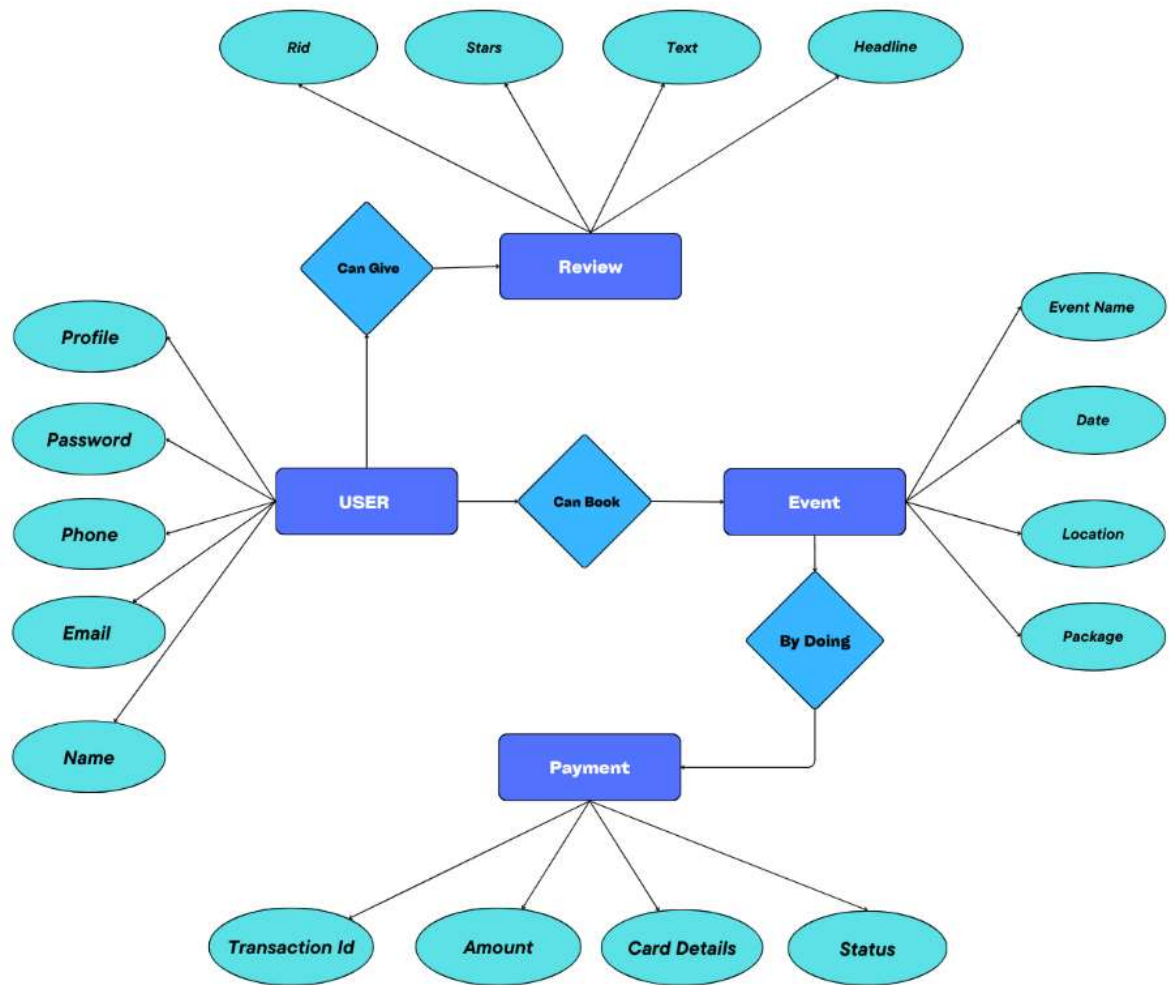


Figure 3.8 ER Diagram

3.3 User Interface Design

The design of the User Interface (UI) for Eventify is in a way that promotes easy use of the platform by the users enabling them to quickly find their way through all the essential features of event management and booking. The illustration depicts the useful structure and the relationship of constituent elements and functional zones of the Eventify platform, including user login, events search, and contact and feedback systems.

Login and Signup – User journey which is the pathway that a user traces on a system begins at the Login/Signup screen where a user chooses to sign up or login by entering a username/email and a password. This measure is put in place to ensure maximum security and that only registered and approved users will be able to use the core functions of Eventify. When the user fills in their details, that information is relayed to the Server in order to cross check with the Database. Those details are verified and using the particulars provided, the user is allowed access to the Homepage.

Homepage and Core Functionalities- The Homepage is the main dashboard that houses all core functionalities and guides users on what to do next including looking for events, reaching out to the event organizers and leaving reviews. The Homepage is also easy to navigate which is very important as it gives users direction on where to go for each function making it possible to complete any task.

Explore Events and Booking- In the Explore Events part, the user can view the Upcoming Events, select any event of interest and continue to book the event. At this point, the selected event is chosen by the users, after which they are given the option to complete the booking process through a Payment gateway for the selected event.

Contact Us – This segment of the website provides a chance to fill in one's details and chat with the organizers on WhatsApp which also promotes the interactivity of the users and organizers. It assists the user's enhancement of additional information regarding the event as well as aiding in the efficient organization of the event.

Review System - The Review section offers users the option to provide feedback by clicking on star ratings and writing reviews. This feature not only allows users to share their experiences but also contributes to maintaining quality and transparency within the platform.

Database and Server - The Database is a critical component, storing all necessary information, including user details, event data, and reviews. The database is closely integrated with the Server to facilitate data validation and ensure secure information management across the platform.

Overall, the Eventify platform's UI design is thoughtfully organized to create a cohesive user experience, prioritizing accessibility, functionality, and interaction secure and interactive environment.

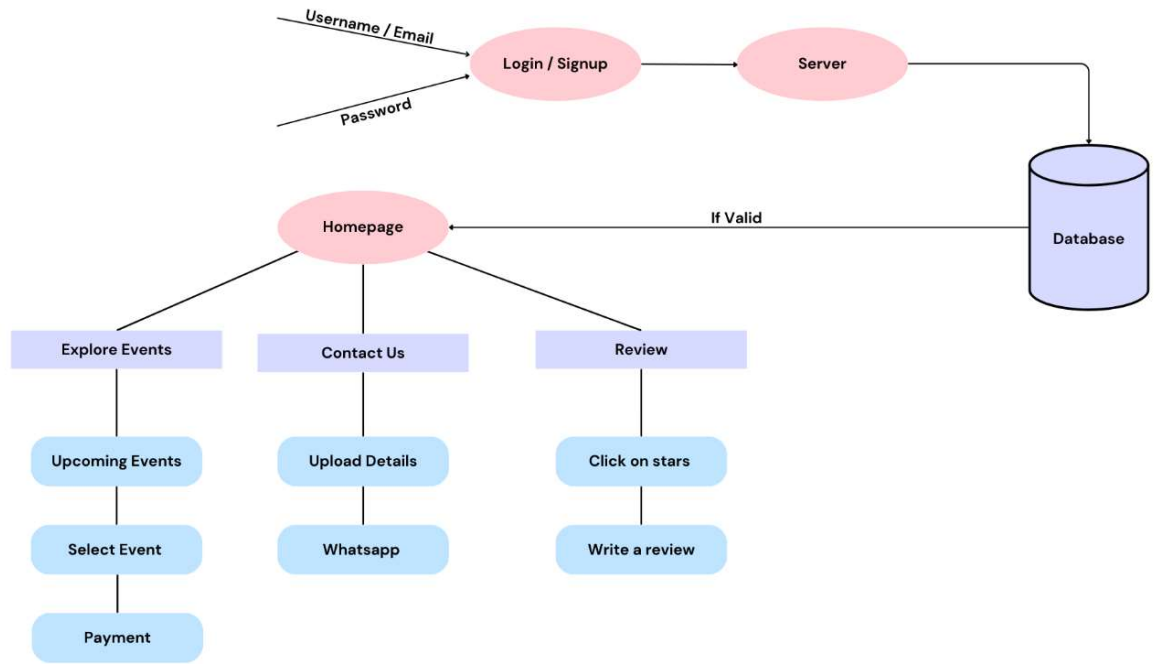


Figure 3.9 User Interface Design

3.4 Methodology

The development of the Eventify platform follows an Agile Software Development Methodology. This approach promotes modular, incremental development, allowing each part of the system to be built, tested, and refined before moving to the next. Agile's structured sprint-based approach supports flexibility, enabling the team to handle complex features like user registration, event management, and secure payment integration efficiently.

1. Agile Methodology in Eventify Development

The Agile process for Eventify is organized into multiple sprints, each dedicated to a specific feature or module. This structured approach includes phases for planning, development, and testing, ensuring each functionality is well-implemented and verified.

- **Sprint Planning:** Each sprint begins with a review of project requirements, prioritizing tasks according to the project roadmap. Early sprints, for instance, focused on implementing foundational features such as user registration and profile management, crucial for platform functionality.
- **Incremental Development:** Functionalities are divided into separate modules (e.g., event creation, booking, payment processing), which are developed independently. This modular approach allows each feature to be built, tested, and refined without affecting other system parts.
- **Testing:** Testing is integrated into each sprint to ensure that each module functions properly. This includes unit tests for isolated components and integration tests to verify seamless interactions, such as the connection between event booking and Google Maps API for venue location.

2. Phases in Agile Development for Eventify

Each Agile sprint in Eventify's development cycle includes four main phases:

- **Requirements Analysis:** For each sprint, requirements for upcoming features are defined in detail, considering both technical constraints and user expectations. For example, in the payment module, security and multi-option payment features were high priorities to ensure a smooth, safe booking experience.
- **Design and Prototyping:** In this phase, initial prototypes for each module are created and reviewed for usability and alignment with project goals.
- **Development:** Each module is developed based on the designs and specifications outlined in previous phases. The MERN stack supports seamless front-end and back-end integration, while APIs like Google Maps and Stripe add crucial functionalities like location and secure payment processing.

- **Testing and Quality Assurance:** Rigorous testing is conducted to identify and resolve any issues within each sprint. Testing includes unit tests for individual components and integration tests to confirm that interconnected features, such as booking confirmation and payment, work smoothly together.

3. Tools and Documentation for Agile Implementation

To support Agile development, the Eventify team uses various tools for tracking progress, documenting changes, and enabling collaboration:

- **Project Management Tools:** Tools like Jira or Trello help manage tasks by assigning responsibilities, setting deadlines, and monitoring progress throughout each sprint.
- **Version Control:** GitHub is used for source code management, allowing for version control, tracking changes, and collaborative development. This setup ensures each update is documented and enables easy rollbacks if necessary.
- **Documentation:** Detailed documentation is maintained for each module, covering functional specifications, design considerations, and code structure. This documentation is essential for understanding each feature and simplifies future maintenance.

Chapter 4: Implementation and Testing

4.1 Introduction to Languages, IDEs, Tools, and Technologies Used

In developing the Carpooling System, a variety of programming languages, frameworks, and tools were used to create an efficient, scalable, and user-friendly platform.

Languages and Frameworks

JavaScript: The core language for both front-end and back-end development, chosen for its seamless compatibility within the MERN stack. JavaScript enables a unified development environment, enhancing efficiency across the application.

React.js: Employed for developing the front-end user interface, React.js uses a component-based architecture that creates a responsive, interactive, and dynamic UI for users, event organizers, and administrators.

Node.js: Functions as the server environment, enabling efficient execution of JavaScript on the server side. Node.js's asynchronous, non-blocking nature makes it well-suited for managing multiple user interactions and real-time event updates.

Express.js: Acts as the back-end framework, simplifying the creation of RESTful APIs for event management, user authentication, payment processing, and feedback handling, providing the necessary infrastructure for all server-side functionalities.

Database

MongoDB: A NoSQL database selected for its flexibility in managing unstructured data, MongoDB stores user, ride, and transaction data in a JSON-like format, which is easily adaptable and scalable.

APIs and Integrations

Stripe: Used as the payment gateway for secure transactions, allowing encrypted payment processing and diverse payment methods.

STEP	DESCIRPTION	SUBTASKS
Start	Begin the payment gateway integration process.	-
1. Set Up Payment Provider Account	Create and configure an account with a chosen payment provider to access payment services.	<ul style="list-style-type: none">- Choose a payment provider (e.g., Stripe, PayPal)- Register an account and complete necessary verifications- Obtain API keys or credentials from the provider
2. Study API Documentation	Review the official API documentation to understand payment processing flows and security guidelines.	<ul style="list-style-type: none">- Review payment processing, authorization, and error handling methods- Study security and compliance requirements for handling payments
3. Implement Payment API in Code	Code the integration, using the payment provider's SDK or API to enable payment functionality.	<ul style="list-style-type: none">- Install required SDK or library- Configure API keys securely in the environment variables

		- Implement payment functionality
End	Complete and finalize the payment gateway integration.	-

Table 4.1 Stripe API Integration Steps

Development Tools

Visual Studio Code (IDE): Used as the primary development environment, offering features like syntax highlighting, code completion, debugging, and integrated terminal support for JavaScript and Node.js.

Git & GitHub: For version control and collaborative development, GitHub provides a platform to track changes, manage different project versions, and ensure smooth collaboration.

4.2 Algorithm/Pseudocode Used

User Registration Algorithm

Purpose: Registers a new user securely and generates a JWT token for session management.

Steps:

1. Extract name, email, password, and phone from the registration request.
2. Check if a user with the provided email already exists in the database:
3. If a user exists, return "User already exists".
4. Validate the email format and check that the password meets strength requirements:
5. If either validation fails, return an appropriate error message.
6. Hash the password using bcrypt for security.
7. Create a new user object with name, email, hashed password, and phone.
8. Save the new user object in MongoDB.

Generate a JWT token for the registered user.

Send a success response with the generated token and "User created successfully".

Pseudocode:

```
function registerUser(name, email, password, phone):  
  
    if isEmailExists(email):  
  
        return "User already exists"  
  
    if not isValidEmail(email) or not isStrongPassword(password):  
  
        return "Invalid email format or weak password"  
  
    hashedPassword = hash(password)  
  
    newUser = createUser(name, email, hashedPassword, phone)  
  
    saveUser(newUser)  
  
    token = generateJWT(newUser)  
  
    return {message: "User created successfully", token: token}
```

2. User Login Algorithm

Purpose: Authenticates an existing user and issues a JWT token upon successful login

Steps:

1. Extract email and password from the login request.
2. Check if a user with the provided email exists in the database:
3. If no user is found, return "User does not exist".
4. Compare the provided password with the stored hashed password:
5. If the passwords do not match, return "Incorrect password".

6. Generate a JWT token for the authenticated user.
7. Send a success response with the generated token and "User logged in successfully".

Pseudocode:

```
function loginUser(email, password):  
  
    if not isEmailExists(email):  
  
        return "User does not exist"  
  
    storedPassword = getStoredPassword(email)  
  
    if not comparePassword(password, storedPassword):  
  
        return "Incorrect password"  
  
    token = generateJWT(email)  
  
    return { message: "User logged in successfully", token: token }
```

4.3 Testing Techniques

The Testing Module is a crucial part of the software development lifecycle for Eventify. Testing ensures that all functionalities work 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. Various testing techniques were applied to ensure the robustness and reliability of the Eventify platform. These include:

Unit Testing

Description: Each component was tested independently to ensure that individual units function correctly. For example, unit tests were created for login functionality, event browsing, data retrieval from MongoDB, and API requests.

Tools Used: Jest was employed for JavaScript testing due to its compatibility with Node.js and React.

Integration Testing

Description: Focused on testing the interactions between different modules. For example, the integration between the front-end event booking form, Stripe payment gateway, and back-end event management was tested to ensure smooth functionality.

Tools Used: Postman was used for API testing, verifying responses from server-side endpoints for event registration, user authentication, and payment processing

Description: System testing was conducted to validate the entire Eventify platform as a whole. The goal was to ensure the end-to-end functionality of the application, including registration, event selection, booking, payment, and review submission.

Tools Used: Manual testing was the primary approach here, simulating complete user flows for comprehensive evaluation.

Usability Testing

Description: Ensured the application's ease of use and user-friendliness. Real users were invited to navigate the system, explore events, complete bookings, and submit feedback on their user experience.

Security Testing

Description: Ensured the security of user data, including personal information and payment details. Tests included SQL injection testing, validation of data encryption, and access control checks to secure all user interactions within Eventify.activity

4.4 Test Cases Designed for the Project

Below is sample test cases designed to cover core functionalities of Eventify.

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”

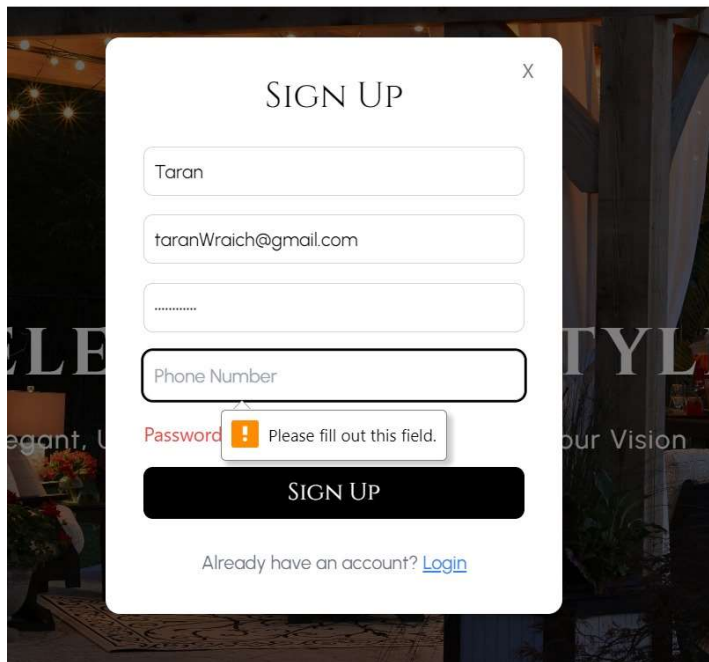
A screenshot of a 'SIGN UP' modal form. The form has a title 'SIGN UP' and a close button 'X' in the top right corner. It contains four input fields: a text field with 'Taran', an email field with 'taranWraich@gmail.com', a password field with masked characters, and a 'Phone Number' field. A red error message 'Please fill out this field.' with an exclamation mark icon is displayed over the 'Phone Number' field. Below the fields is a 'SIGN UP' button and a link 'Already have an account? Login'. The background is a blurred image of an event space.

Figure 4.1 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 enter a valid email address.”.

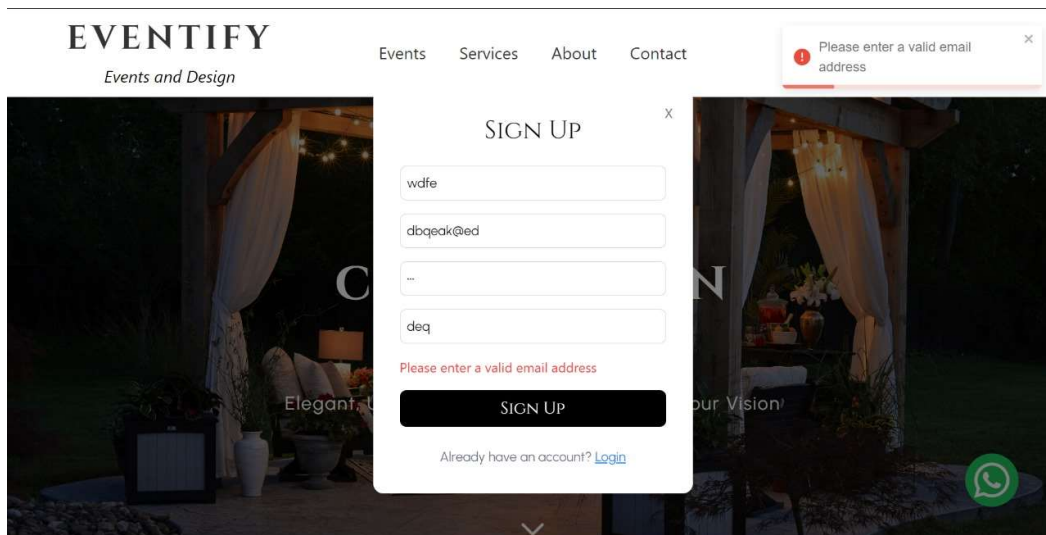


Figure 4.2 Required valid email

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."

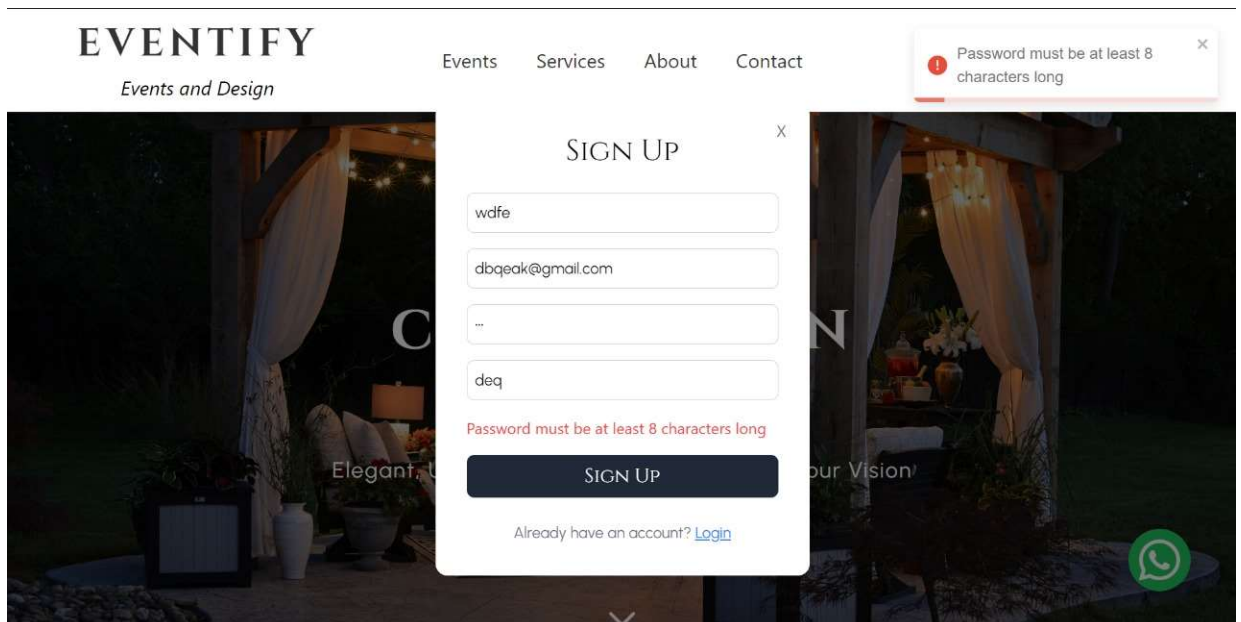


Figure 4.3 password must be at least 8 character long

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".

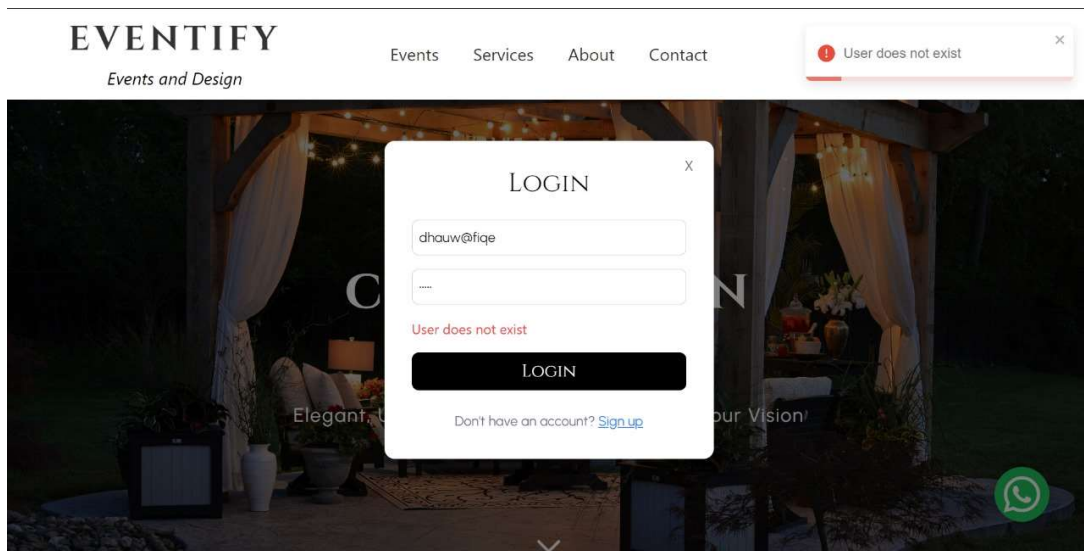


Figure 4.4: Invalid login credentials

Test Case 5: This test case verify that the payment process fails when invalid card credentials are provided.

Input: Payment Details: Invalid card number, expiration date, or CVV.

Expected Output: Error message, "Payment could not be completed. Your payment didn't go through as it was declined by the bank. Try another payment method or contact your bank."

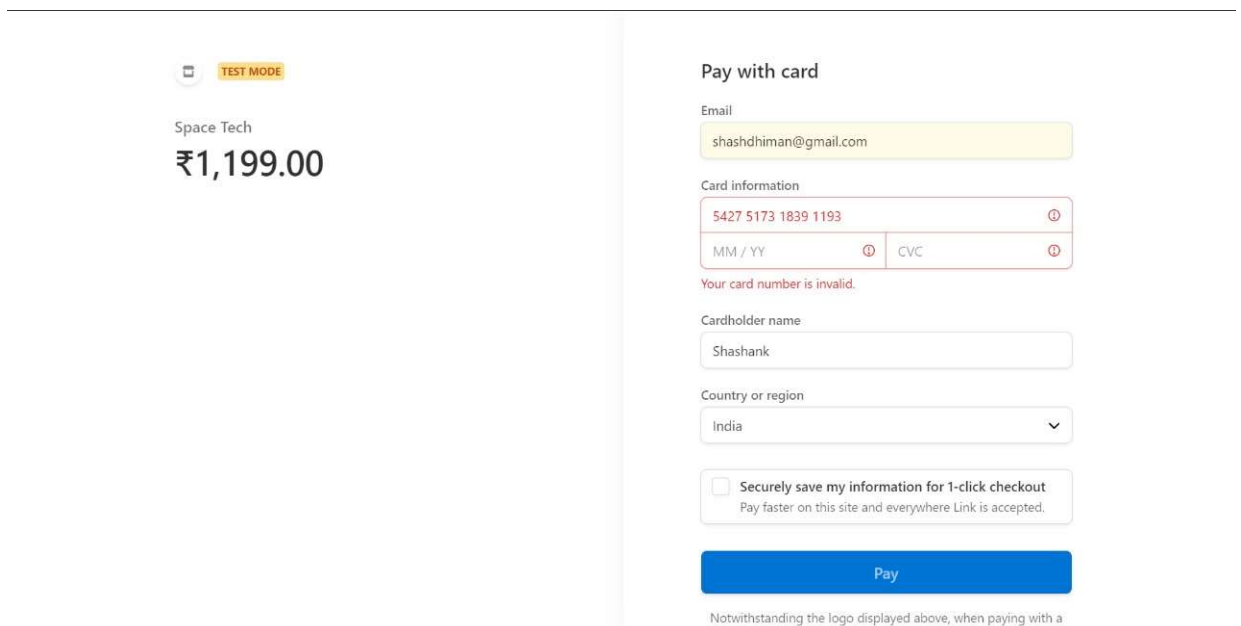


Figure 4.5 Payment Processing with Invalid Card Credentials

Test Case 6: This test case tests the Stripe payment form in test mode for an event costing ₹1,199. Fields include email, card information, cardholder name, and country. Validates required fields and payment processing.

Expected Result: All fields should validate correctly, and payment should process successfully in test mode. Missing fields should trigger error messages.

The screenshot shows a Stripe payment form in test mode. On the left, it displays 'Space Tech' and the amount '₹1,199.00'. The main form is titled 'Pay with card' and includes the following fields:

- Email:** A text input field with a red 'REQUIRED' label.
- Card information:** A section with a red 'REQUIRED' label, containing:
 - Card number: A text input field with the placeholder '1234 1234 1234 1234'.
 - MM / YY: A text input field for the expiration month and year.
 - CVC: A text input field for the card verification code.
- Cardholder name:** A text input field with a red 'REQUIRED' label and the placeholder 'Full name on card'.
- Country or region:** A dropdown menu with 'India' selected.
- Securely save my information for 1-click checkout:** A checkbox with the text 'Pay faster on this site and everywhere Link is accepted.'
- Pay:** A blue button.

At the bottom, there is a disclaimer: 'Notwithstanding the logo displayed above, when paying with a co-branded eftpos debit card, your payment may be processed through either card network.'

Figure 4.6 Required field during payment

Test Case 7: This test case tests the review submission functionality. The form allows users to rate with stars, enter their first and last names, a brief title, and a detailed review.

Expected Result: All fields should be filled correctly. If any field is empty, an error message should prompt the user to complete it (e.g., "Please fill out this field."). Upon successful submission, the review should be saved and displayed.

Result: Pass

Write a Review

Click on star to review

★ ★ ★ ★ ★

Anshika

Sharma

Awesome work by team!

Write your Review

!

Please fill out this field.

Figure 4.7 Review submission form

Chapter 5: Results and Discussions

5.1 User Interface Representation

The Eventify platform features an intuitive user interface (UI) designed to enhance the experience for users and organizers alike. The UI is structured to provide a seamless, user-friendly experience across various modules, allowing users to register, book events, manage profiles, and process payments efficiently.

5.1.1 Brief Description of Various Modules of the System

User Authentication and Profile Management Module

Description: Provides user registration and login functionality. This module includes profile management where users can update personal information, upload verification documents, and manage preferences. Key Features: Secure login, password hashing, document upload, and profile editing.

Event Management Module

Description: Allows users to search for events, book tickets, and view event details. Organizers can create and manage events by adding relevant details such as location, date, ticketing options, and pricing.

Key Features: Search bar with Google Maps integration for venue location, date selection, event filters (e.g., type, availability, price range), and booking functionality.

Payment Processing Module

Description: Handles secure payments between users. The system utilizes a third-party payment gateway like Stripe to enable encrypted, reliable transactions. Key Features: Payment gateway integration, fare calculation, payment confirmation, and transaction records.

5.2 Snapshots of System with Brief Detail of Each and Discussion

The following snapshots illustrate the main user interface components and functionality of the Eventify system.

User Interface

The User Interface of the Eventify platform is designed to deliver a seamless and user-friendly experience for both event organizers and attendees. It enables users to easily register, log in, and manage their profiles, including updating personal information

The interface supports event management by allowing attendees to search for available events and book events. User can also book the already registered events and make payment for it.

The interface is intuitive, providing a visually organized layout with easy navigation, making event discovery, booking, and management effortless for all users.

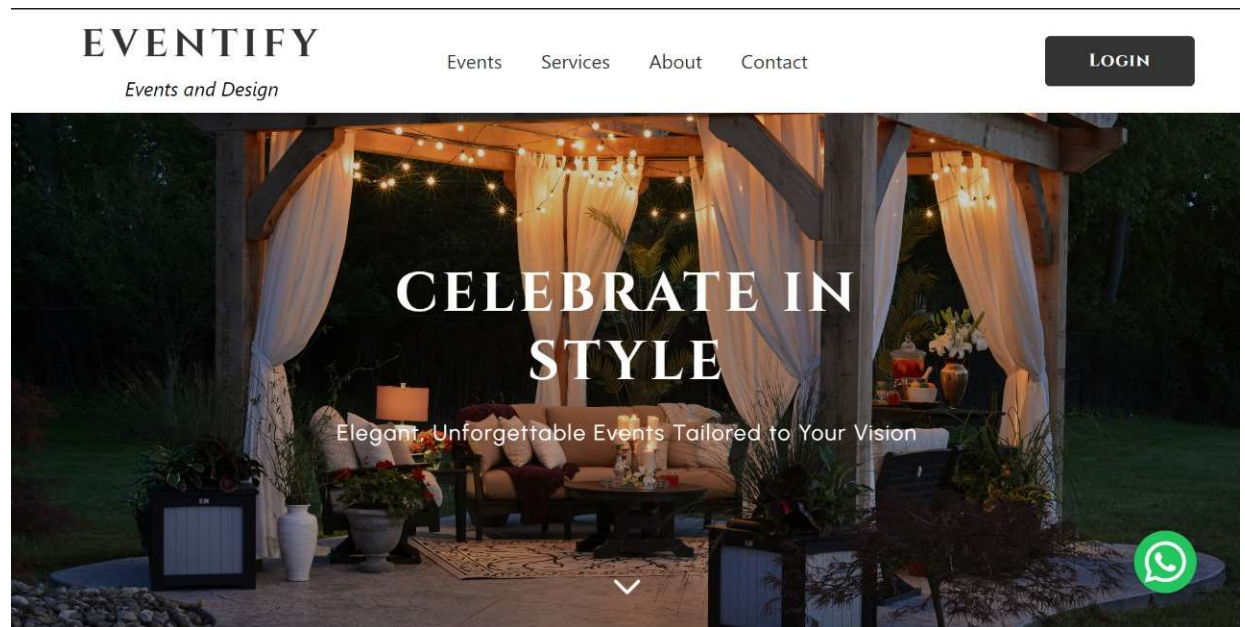


Figure 5.1 User Interface

User Registration and Login Interface

The user sign-up interface allows new users to register by entering their personal details, creating a secure account to access event booking features on the platform.

The screenshot displays the Eventify website's user registration interface. The header features the 'EVENTIFY' logo with the tagline 'Events and Design', navigation links for 'Events', 'Services', 'About', and 'Contact', and a 'LOGIN' button. The background image shows an outdoor event space with a gazebo and string lights. A 'SIGN UP' modal form is centered on the screen, featuring input fields for 'Username', 'Email', 'Password', and 'Phone Number', a 'SIGN UP' button, and a link for users who already have an account.

Figure 5.2 User Sign up

The login interface enables existing users to access their accounts securely by entering their registered email and password, ensuring authenticated access to the platform.

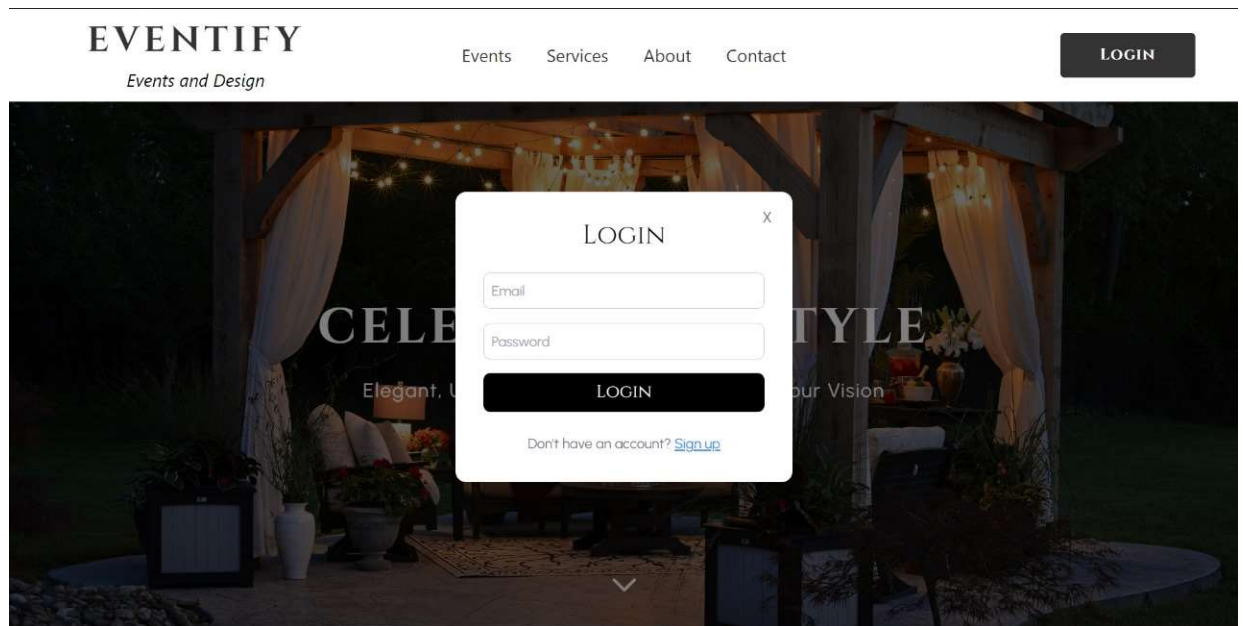


Figure 5.3 User Login

Exploring and Booking Event Interface

This interface will help user to explore the already existing events and the user can organize their own event of different category.

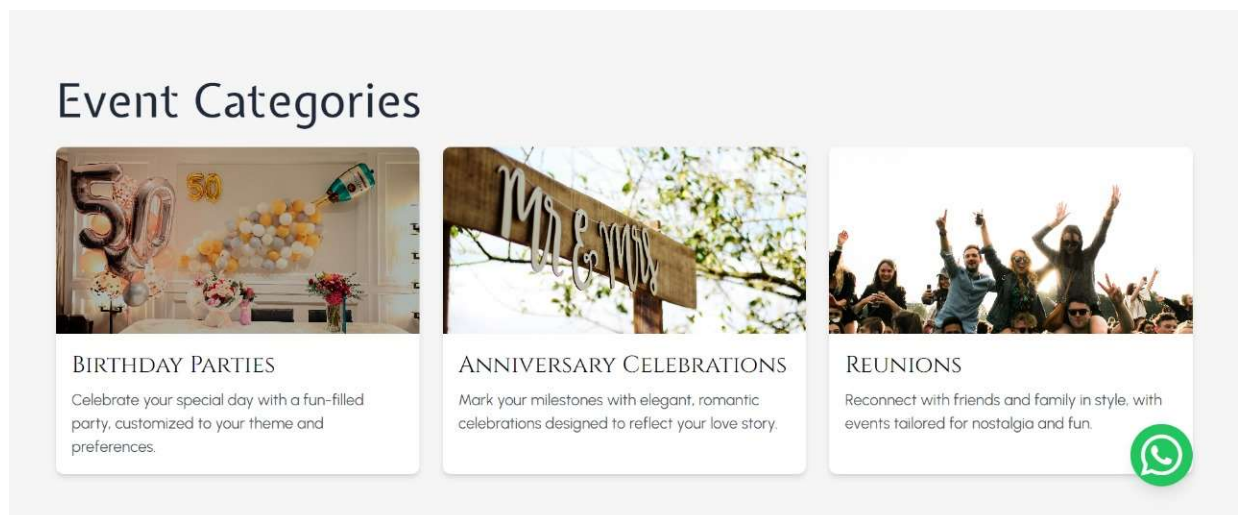


Figure 5.4 Event Categories

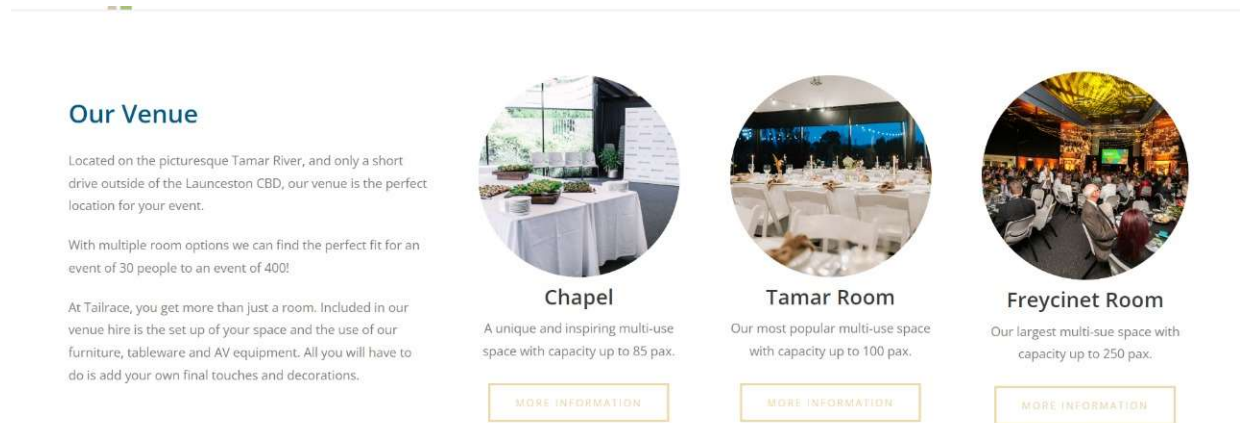


Figure 5.5 Our venues

Adding Review Interface

This interface is designed to allow the user to give reviews about the event organized by Eventify.

It helps the new user to know better about the functioning of Eventify.

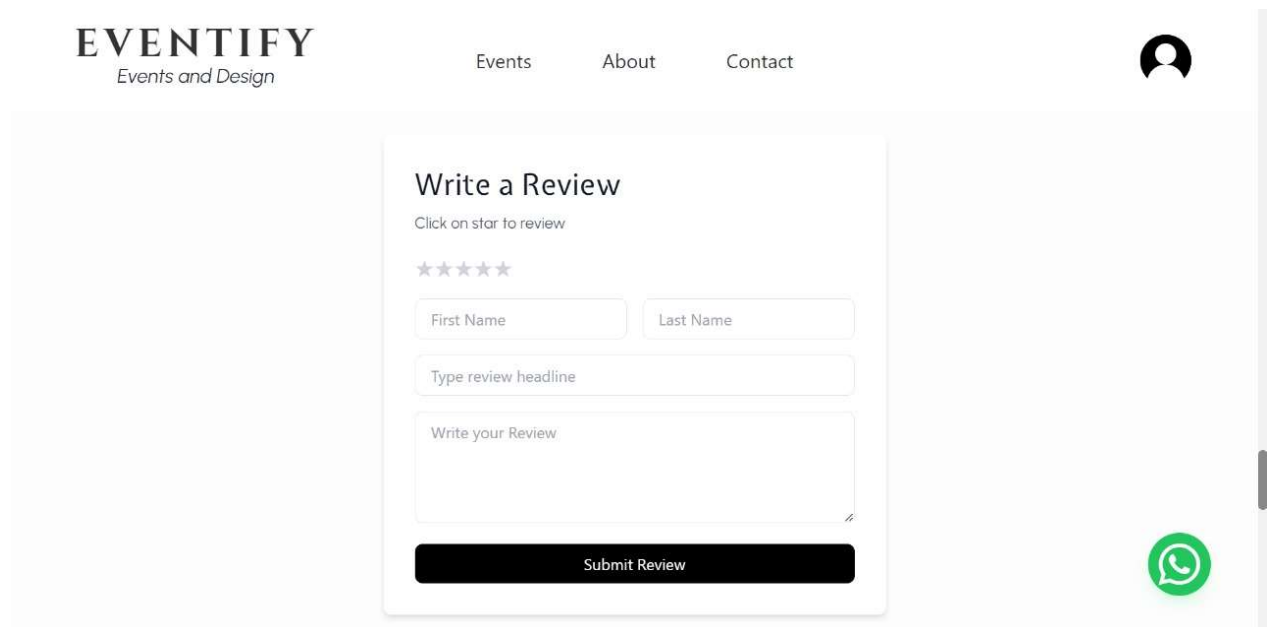


Figure 5.6 Adding Review Interface

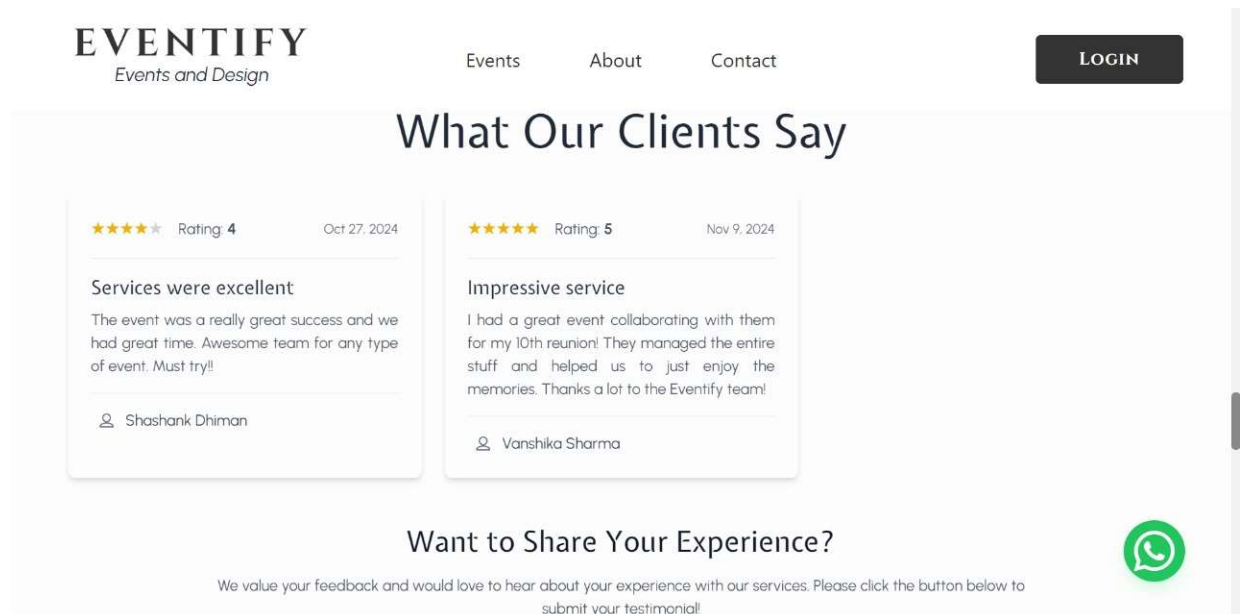


Figure 5.7: Review Section

Payment Processing Interface

The screenshots showcase the payment processing interface for a platform called "Eventify," secured by Stripe. The user is presented with a price summary, displayed in the top left, indicating an amount to be paid. Various payment options are available, including UPI, cards, net banking, wallets, and a "Pay Later" option.

In the first screenshot, a UPI QR code is provided for quick payment, with recommended options displayed below for a personalized experience.

The second screenshot confirms a successful transaction, showing a green screen with a checkmark indicating payment completion. The transaction details, including the amount (₹1), time, and payment method (net banking) are displayed, along with a link for further support from Razorpay.

TEST MODE

Christmas Eve

₹999.00

Pay with card

Email

shashdhiman@gmail.com

Card information

4242 4242 4242 4242

VISA

12 / 34

CVC

Cardholder name

Shashank

Country or region

India

☐ Securely save my information for 1-click checkout
Pay faster on this site and everywhere Link is accepted.

Pay

Notwithstanding the logo displayed above, when paying with a co-branded eftpos debit card, your payment may be processed through either card network.

Figure 5.8 Payment Processing Interface

TEST MODE

Diwali Parties

₹1,499.00

✓

Thanks for your payment

A payment to Stripe will appear on your statement.

STRIPE ₹1,499.00

Powered by stripe | Terms | Privacy

Figure 5.9 Payment Successful

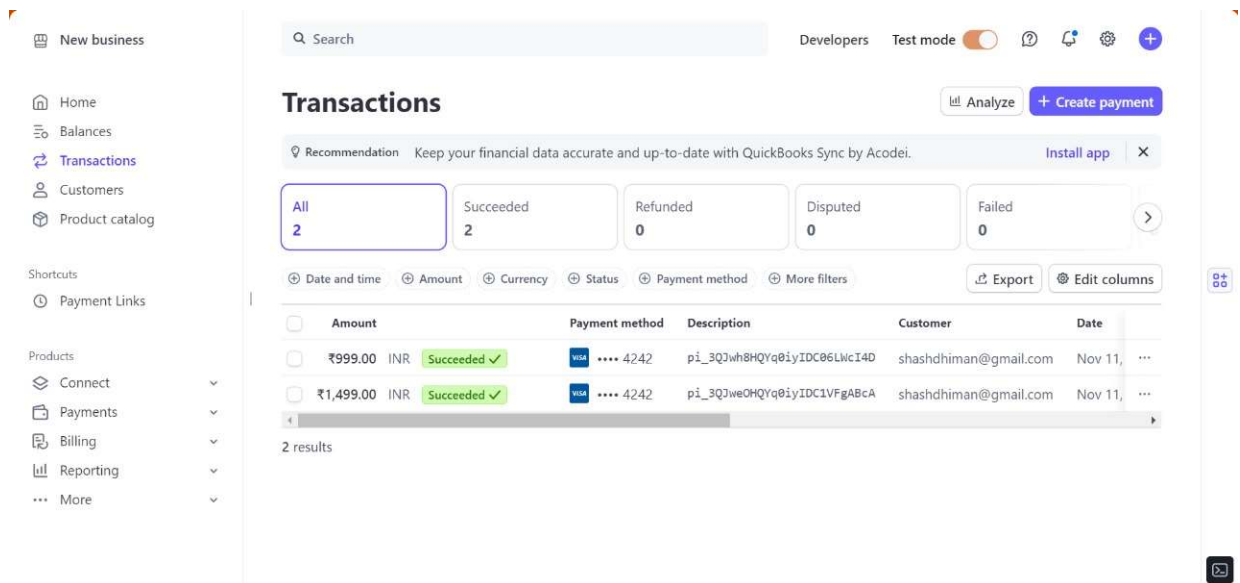


Figure 5.10: Transaction history

5.3 Back-End Representation (Database)

The Eventify System relies on MongoDB to store and manage user data, information, transaction details, and other core system data. MongoDB's document-based structure allows for flexibility, supporting rapid retrieval and storage of dynamic data.

5.3.1 Snapshots of Database Tables with Brief Description

Below are key database collections and their structure:

User Collection

Stores all user-related data, including login credentials and profile information. The `isVerified` field indicates whether a user is registered or not.

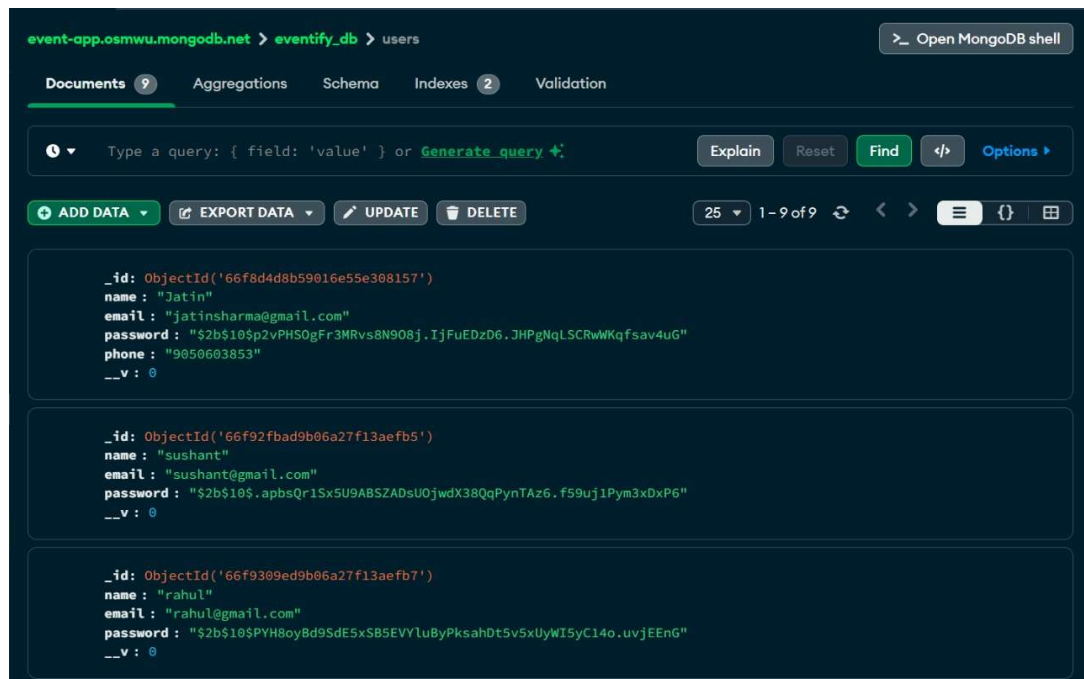


Figure 5.11 User Collection Database

Review section Collection

Manages details for each ride, such as the route, available seats, and fare per seat. The rideStatus field indicates the current status (e.g., available, booked, completed).

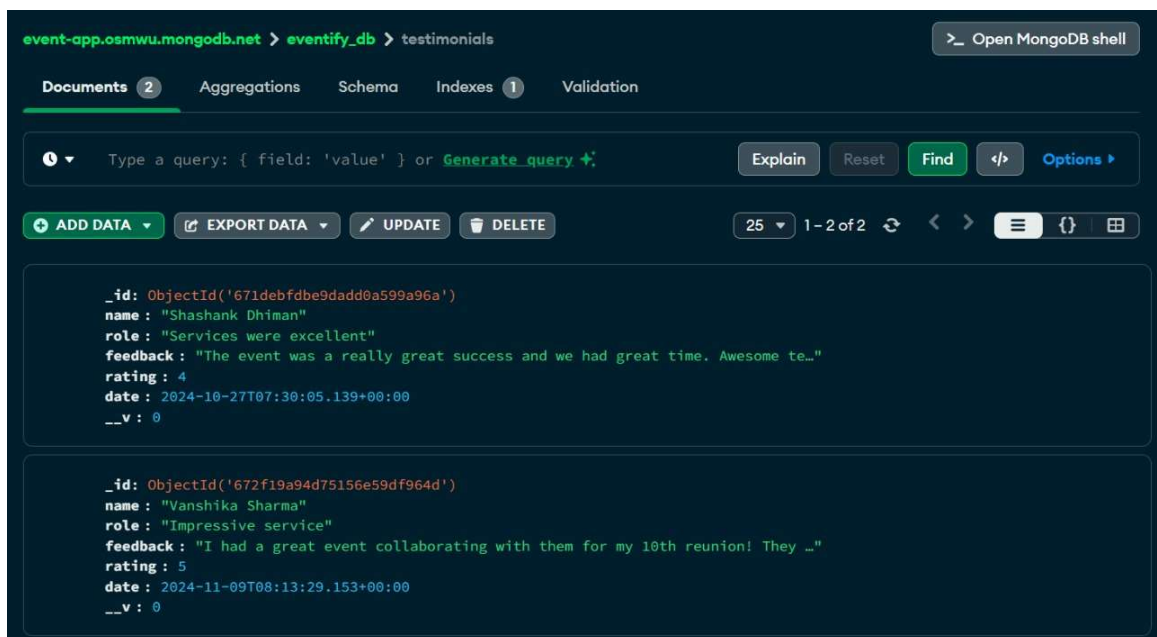


Figure 5.12 Review Section - testimonial Database

Feedback Form

Stores the data and information to give answer to the queries asked in the contact US form.

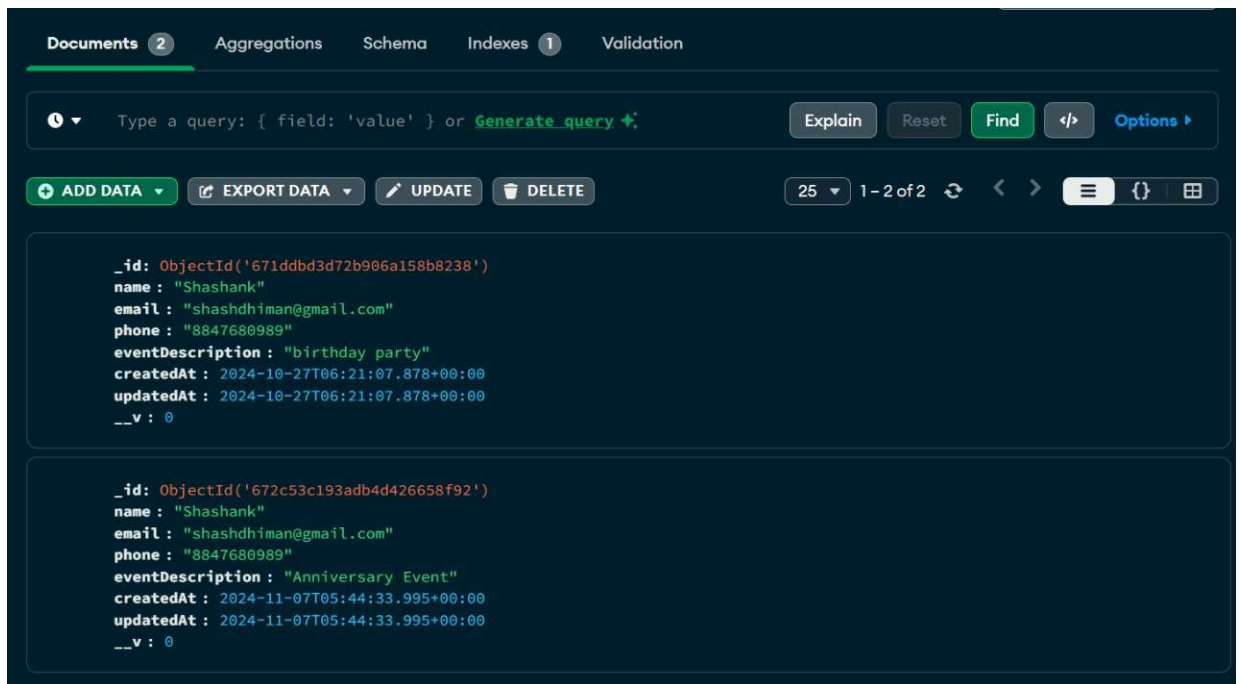


Figure 5.13 Feedback Database

Chapter 6: Conclusion and Future Scope

6.1 Conclusion

The Eventify platform has now been able to realize its main target goals as that of ensuring effective and simpler event booking system for the users. With the help of MERN stack and using a dependable payment processor, Eventify takes care of the many issues that are inherent within the old ways of event management systems. The platform illustrates the potential of new webs in transforming the way users interact with events by providing features such as easier event search, real time booking, and updates.

- **Improved Client Protection:** The implemented user authentication, comprehensive profile confirmation, and the approval of documents provided by the organizers guarantee a trustworthy atmosphere to accelerate confidence in respect of the organizers to the audience and vice versa.
- **Convenient Event Creation:** Attendees are able to find and reserve events with ease by employing complex searches, whilst event holders have an event publishing system that is efficient even in the presence of Google maps for accurate geolocation.
- **Integration of a Secure Payment Gateway:** By ensuring that all ticket transactions and purchases are done through a safe gateway, there is assurance that channels of exchange are secure and effective.
- **System for User Ratings:** The feature for ratings and reviews provides chance for accountability and feedback to the audience where they feel free to express themselves and assist the organizers to know how to do better next time.

6.2 Future Scope

Eventify has achieved its objectives that were first set out for this application, however, there is a host of potential variables that can be value added in such a way that development and additional features can be added. Some potential improvement areas are as follows:

Mobile Application Development: Creating native applications for both iOS and Android platforms would greatly improve usability and engagement. A mobile app would also assist in accessing real time updates, notifications as well as booking confirmations and focusing on a wider audience.

In-App Messaging System: If an in-app messaging system is implemented, then attendees and organizers will be able to contact one another through the app. This will enable them to finalize bookings, discuss the events, and plan logistics without employing any third-party messaging applications.

Dynamic Pricing Model: A dynamic pricing model for tickets whereby the price displayed would be influenced by many factors such as demand, how many seats are available for the particular event, and even how popular the event is, can enable a more marketable model and also an attractive one for attendees and organizers.

Real-Time Event Tracking: If tracking of geographical locations of event sites and even the doorways of events bodies and even specific places were made possible, attending events would be made easier since details of where to present oneself will be provided and even a change in the event information will be communicated.

If these authorities are able to perform consistently then Eventify can also see an enhancement in its functions and an improvement in the user experience as well.

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.

Appendix A: Development Environment

1. Programming Languages and Frameworks

- JavaScript: Primary language for both front-end and back-end development, ensuring seamless integration across the MERN stack.
- React.js: Front-end library used for creating dynamic, component-based user interfaces.
- Node.js: Server-side environment that allows JavaScript to be run on the server, supporting asynchronous, real-time operations.
- Express.js: Back-end web application framework used for building APIs, handling HTTP requests, and routing.
- MongoDB: NoSQL database for managing user, ride, and payment data with a flexible, JSON-like structure.

2. Integrated Development Environment (IDE)

- Visual Studio Code: A widely-used IDE with features like syntax highlighting, code debugging, Git integration, and extensions tailored for JavaScript, React, and Node.js development.

3. Version Control System

- Git & GitHub: Version control tools used to manage code changes, track project progress, and facilitate collaboration. GitHub provided an online platform for code storage, issue tracking, and pull requests, essential for a smooth development workflow.

4. APIs and Integrations

- Stripe: Payment gateway API integrated to enable secure, encrypted transactions within the platform, supporting various payment methods like UPI and credit/debit cards.

5. Development and Testing Tools

- Postman: API testing tool for verifying server responses and debugging API endpoints.
- MongoDB Compass: Cloud-hosted MongoDB service for database management, offering features like automated backups, data encryption, and performance monitoring.

6. Hosting and Deployment

- MongoDB Compass: Cloud-hosted database solution for reliable, scalable data management.

This development environment, combined with Agile practices, provided a robust foundation for building and testing the platform's features, ensuring scalability, security, and user-friendliness throughout the project lifecycle.