**A**

***PROJECT REPORT***

*on*

**“EVENT BOOKING SYSTEM USING MERN STACK”**

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

**BACHELOR OF TECHNOLOGY**

****

Session: - Jan-June 2023

Submitted by

Ganesham Tailor (19ETCCS021)

Shreya (19ETCCS066)

Vishwajeet Singh (19ETCCS081)

VIII semester, CSE

Under Guidance of

Aaditya Maheshwari

Head of Industry Project

CSE & Techno India NJR Institute of Technology

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**TECHNO INDIA NJR INSTITUTE OF TECHNOLOGY, UDAIPUR-313001**

**MAY – 2023**

**A**

***PROJECT REPORT***

*on*

**“EVENT BOOKING SYSTEM USING MERN STACK”**

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

**BACHELOR OF TECHNOLOGY**



**Session: - Jan-June 2023**

Under Guidance of

Aaditya Maheshwari

Head of Industry Project

CSE & Techno India NJR Institute of Technology

Submitted by

Ganesham Tailor(19ETCCS021)

Shreya (19ETCCS066)

Vishwajeet Singh(19ETCCS081)

VIII semester, CSE

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**TECHNO INDIA NJR INSTITUTE OF TECHNOLOGY, UDAIPUR-313001**

**MAY – 2023**



Department of Computer Science and Engineering

Techno India NJR Institute of Technology, Udaipur-313001

**Certificate**

This is to certify that project work titled “EVENT BOOKING SYSTEM USING MERN STACK” by **Ganesham Tailor** successfully carried out in the Department of Computer Science and Engineering, TINJRIT and the report is approved for submission in the partial fulfilment of the requirements for award of degree of Bachelor of Technology in Computer Science and Engineering.

Aaditya Maheshwari Dr. Rimpy Bishnoi

Head of Industry Project Head of Department

CSE, Techno India NJR Institute of Technology Dept. of CSE TINJRIT, Udaipur

Date...................... Date......................



Department of Computer Science and Engineering

Techno India NJR Institute of Technology, Udaipur-313001

**Certificate**

This is to certify that project work titled “EVENT BOOKING SYSTEM USING MERN STACK” by **Shreya** successfully carried out in the Department of Computer Science and Engineering, TINJRIT and the report is approved for submission in the partial fulfilment of the requirements for award of degree of Bachelor of Technology in Computer Science and Engineering.

Aaditya Maheshwari Dr. Rimpy Bishnoi

Head of Industry Project Head of Department

CSE, Techno India NJR Institute of Technology Dept. of CSE TINJRIT, Udaipur

Date...................... Date......................



Department of Computer Science and Engineering

Techno India NJR Institute of Technology, Udaipur-313001

**Certificate**

This is to certify that project work titled “EVENT BOOKING SYSTEM USING MERN STACK” by **Vishwajeet Singh** successfully carried out in the Department of Computer Science and Engineering, TINJRIT and the report is approved for submission in the partial fulfilment of the requirements for award of degree of Bachelor of Technology in Computer Science and Engineering.

Aaditya Maheshwari Dr. Rimpy Bishnoi

Head of Industry Project Head of Department

CSE, Techno India NJR Institute of Technology Dept. of CSE TINJRIT, Udaipur

Date...................... Date......................



Department of Computer Science and Engineering

Techno India NJR Institute of Technology, Udaipur-313001

**Examiner Certificate**

This is to certify that the following student

**Ganesham Tailor**

of final year B.Tech. (Computer Science and Engineering), was examined for the project work titled

***“Event Booking System Using MERN”***

during the academic year 2022 – 2023 at Techno India NJR Institute of Technology, Udaipur

**Remarks:**

**Date:**

 Signature Signature

 (**Internal Examiner**) (**External Examiner**)

Name :- ……………………… Name :- ………………………

Designation:- ……………….. Designation:- ………………..

Department: - ………………. Department: - ……………….

Organization:- ……………… Organization:- ………………



Department of Computer Science and Engineering

Techno India NJR Institute of Technology, Udaipur-313001

**Examiner Certificate**

This is to certify that the following student

**Shreya**

of final year B.Tech. (Computer Science and Engineering), was examined for the project work titled

***“Event Booking System Using MERN”***

during the academic year 2022 – 2023 at Techno India NJR Institute of Technology, Udaipur

**Remarks:**

**Date:**

 Signature Signature

 (**Internal Examiner**) (**External Examiner**)

Name :- ……………………… Name :- ………………………

Designation:- ……………….. Designation:- ………………..

Department: - ………………. Department: - ……………….

Organization:- ……………… Organization:- ………………



Department of Computer Science and Engineering

Techno India NJR Institute of Technology, Udaipur-313001

**Examiner Certificate**

This is to certify that the following student

**Vishwajeet Singh**

of final year B.Tech. (Computer Science and Engineering), was examined for the project work titled

***“Event Booking System Using MERN”***

during the academic year 2022 – 2023 at Techno India NJR Institute of Technology, Udaipur

**Remarks:**

**Date:**

 Signature Signature

 (**Internal Examiner**) (**External Examiner**)

Name :- ……………………… Name :- ………………………

Designation:- ……………….. Designation:- ………………..

Department: - ………………. Department: - ……………….

Organization:- ……………… Organization:- ………………

**Preface**

An event booking system serves as a digital gateway, providing a centralized platform for seamless event registration, booking, and managing events and tours. It leverages the power of technology to simplify the entire event lifecycle, from event creation and promotion to booking sales and check-in.

Event organizers can utilize an event booking system to efficiently manage event details, track registrations, and gain valuable insights into users preferences. It automates tasks such as ticket generation, payment processing, and communication, enabling organizers to focus on creating exceptional event experiences.

For attendees, an event booking system offers a convenient and user-friendly interface to explore a wide range of events, access event details, and securely purchase tickets. It eliminates the need for physical ticket collection, reduces waiting times, and provides real-time updates and notifications.

The benefits of an event booking system are numerous. It streamlines the entire event management process, increases event visibility, expands audience reach, and enhances the overall attendee experience. It empowers event organizers with powerful tools to plan, promote, and execute successful events, while providing attendees with a hassle-free and enjoyable event booking journey.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

TECHNO INDIA NJR INSTITUTE OF TECHNOLOGY, UDAIPUR-313001

**ACKNOWLEDGMENT**

We take this opportunity to record our sincere thanks to all who helped us to successfully complete this work. Firstly, We are grateful to our **Aaditya Maheswari** for his invaluable guidance and constant encouragement, support and most importantly for giving us the opportunity to carry out this work.

We would like to express our deepest sense of gratitude and humble regards to our

**Head of Department Dr. Rimpy Bishnoi** for giving invariable encouragement in our endeavours and providing necessary facility for the same. Also a sincere thanks to all faculty members of CSE, TINJRIT for their help in the project directly or indirectly.

Finally, We would like to thank my friends for their support and discussions that have proved very valuable for us. We are indebted to our parents for providing constant support, love and encouragement. We thank them for the sacrifices they made so that we could grow up in a learning environment. They have always stood by us in everything we have done, providing constant support, encouragement and love

**Ganesham Tailor (19ETCCS021)**

**Shreya (19ETCCS066)**

**Vishwajeet Singh (19ETCCS081)**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

TECHNO INDIA NJR INSTITUTE OF TECHNOLOGY, UDAIPUR-313001

 **CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Abstract** | …………………………………….………………………… | i |
| **Contents** | ………………………………………………………………. | ii – iii |
| **List of Figures** …………………………………………………….............. | iv |
| **List of Abbreviations**……………………………………………………… | vi  |
| **Chapter 1** | INTRODUCTION…………………………………………... | 7 |
|  | * 1. Introduction to Event Booking…………...…………..….
 | 7 |
|  | 1.2 Functionalities ………….…………....…………………. | 8 |
|  | 1.2.1 Online Booking………..………..………….…………. | 8 |
|  | 1.2.2 Tour Details and Description….……………………....1.2.3 Event Search and Discovery..…………..………...…....1.2.4 Availability ….…………..………………………….....1.2.5 Reviews and Ratings………………………………….. | 8889 |
| **Chapter 2** | MERN STACK………….………………………………….. | 10 |
|  | * 1. About MERN Stack…………….………….…………...
 | 10 |
|  | * 1. MongoDB………….………...…………………………
 | 13 |
|  | * 1. Express…………..……..………………………………
 | 14 |
|  | * 1. React…………..……………………………………….
 | 15 |
|  | * 1. NodeJS………………………………………………….
 | 16 |
| **Chapter 3** | SYSTEM ANALYSIS AND DESIGN………………………... | 18 |
|  | * 1. Use Case Diagram……………...…………...…………..….
 | 18 |
|  | 3.2 Context Diagram ………….…………....…………………. | 19 |
|  | 3.3 Activity Flow Diagram.….....………..………….…………. | 20 |
| **Chapter 4** | PROJECT PROFILE ..…….…………………………………... | 23 |
|  | * 1. Login Page……………………….………….………….......
 | 23 |
|  | * 1. Register Page………….………...…………………………
 | 24 |
|  | * 1. Booking Section……..……..………………………………
 | 25 |
|  | * 1. Review Section...…..……………………………………….
 | 26 |
|  | * 1. Database...………………………………………………….
 | 28 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| Fig 1 |  MERN Stack………….…………………………………....... | 11 |
| Fig 2 | ER Diagram……….………………………………………….. | 13 |
| Fig 3 | Use Case Diagram …………………………………………….. | 19 |
| Fig 4 | Context Diagram……………………………………………… | 20 |
| Fig 5 | Activity Flow Diagram ……………………………………….. | 21 |
| Fig 6 | Login Page…………….………………………………………. | 23 |
| Fig 7 | Register Page………………………………………………....... | 24 |
| Fig 8 | Booking Section…… ………………………………………..... | 26 |
| Fig 9 | Review Section…………………………………………………  | 27 |
| Fig 10 | MongoDB Database………….…………………………........ | 28 |

**List of Abbreviations/ Symbols**

|  |  |
| --- | --- |
| HTTP | Hypertext Transfer Protocol |
| HTML | Hypertext Markup language |
| CSS | Cascading Style Sheet |
| NPM | Node Package Manager |
| NVM | Node Version Manager |
| JSX | JavaScript XML |
| JSON | JavaScript Object Notation |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**CHAPTER 1 : INTRODUCTION**

* 1. **Introduction to Event Booking:**

An event booking system offers convenience to both event organizers and attendees. It eliminates the need for manual processes such as phone calls or in-person registrations, allowing users to book events anytime, anywhere, and at their convenience. It simplifies the entire booking process and saves time and effort for all parties involved.

Overall, an event booking system enhances the booking experience for both organizers and attendees, improves operational efficiency, and provides valuable data for event management and analysis. It is an essential tool for streamlining event processes and ensuring a smooth and successful event booking experience.

With an event booking system, users can access real-time information about event availability, ticket prices, seat availability, and event details. This allows them to make informed decisions and secure their spots promptly. The system ensures accurate and up-to-date information is available to users, reducing the chances of overbooking or misinformation.

An event booking system can handle a large volume of bookings and registrations without any issues. It is scalable and can accommodate events of varying sizes and complexities. Whether it's a small conference or a large-scale music festival, the system can efficiently manage the booking process, handle ticket sales, and handle attendee data without experiencing performance limitations.

* 1. **Functionalities:**

**1.2.1 Online Booking**

The system enables users to book tours online with a simple and secure booking process. Users can select their desired tour, choose the number of participants, and make a reservation by providing necessary details such as names, contact information, and any specific requirements. The system may offer different payment options, including credit/debit card, online wallets, or bank transfers.

**1.2.2 Tour Details and Descriptions**

Each tour listing includes comprehensive information such as itinerary, activities, accommodations, transportation, inclusions, exclusions, and pricing. Users can access detailed descriptions, photos, and reviews to make informed decisions about the tours they want to book.

**1.2.3 Event Search and Discovery**

The system allows users to search for tours based on their desired destination, date, duration, and other preferences. It provides a user-friendly interface where users can explore various tour options and discover new destinations or experiences.

**1.2.4 Availability**

The booking system provides real-time availability updates for each tour. Users can see the number of available spots, booked seats, and any upcoming departures. This helps users plan their travel accordingly and avoid any conflicts or overbooking.

**1.2.4 Reviews and Ratings:**

Many tour booking systems allow users to leave reviews and ratings for the tours they have experienced. This helps future users make informed decisions based on the feedback and experiences shared by others.

**1.3. Need of Event Booking:**

An event booking site allows organizers to provide comprehensive information about the event, including schedules, venues, speakers, agendas, and any special instructions. Attendees can access this information easily, making informed decisions about which events to attend.

An event booking site provides real-time availability updates on event tickets, seat allocations, and other relevant details. Attendees can instantly see which events have available slots, making it easier for them to secure their spot without any delays or misunderstandings.

By having an online presence, an event booking site ensures that potential attendees can easily access event information, browse through available options, and book tickets or registrations. This accessibility expands the reach of events to a broader audience, potentially increasing attendance.

An event booking site simplifies the event registration and booking process for attendees. It eliminates the need for manual registration forms, phone calls, or in-person visits, allowing users to conveniently book events online from anywhere, at any time.

**CHAPTER 2 : MERN STACK**

**2.1. About MERN Stack:**

The MERN stack is a popular web development stack that consists of four key technologies: MongoDB, Express.js, React, and Node.js. Here's an overview of each component:

1. MongoDB: As mentioned earlier, MongoDB is a NoSQL database that stores data in JSON-like documents.
2. Express.js: Express.js is a web application framework for Node.js. It provides a robust set of features for building web APIs and handling HTTP requests and responses.
3. React: React is a JavaScript library for building user interfaces. It allows you to create reusable UI components and efficiently manage state and rendering.
4. Node.js: Node.js is a server-side JavaScript runtime environment. It allows you to run JavaScript code outside of a web browser, making it suitable for server-side development.

**The MERN stack** combines these technologies to create a full-stack JavaScript development environment. MongoDB serves as the database, Express.js handles the backend server and routing, React handles the frontend UI components and state management, and Node.js provides the runtime environment for server-side code.



 Fig 1 – MERN Stack

**2.2. MongoDB:**

**MongoDB Atlas** is a fully managed cloud database service provided by MongoDB. It offers a convenient and scalable way to deploy, manage, and scale MongoDB databases in the cloud. With MongoDB Atlas, developers can focus on building their applications without worrying about the underlying infrastructure and database administration tasks.

One of the key advantages of MongoDB Atlas is its ease of use. It provides a user-friendly web interface and a well-documented API that simplifies the process of setting up and managing MongoDB databases. Creating a new cluster is straightforward, and Atlas offers various configuration options to tailor the database deployment to specific needs, such as choosing the cloud provider, region, and instance size.

Scalability is another significant benefit of MongoDB Atlas. As application demands grow, Atlas allows for seamless horizontal scaling by adding more nodes to the cluster or using sharding techniques to distribute data across multiple servers. This ensures that the database can handle increased workloads and provide high availability and performance.

MongoDB Atlas also provides built-in data security features. It offers various authentication methods, including username/password, certificate-based authentication, and integration with external authentication providers like LDAP and Active Directory. Encryption at rest and in transit ensures that data remains secure, and role-based access controls enable fine-grained control over database access.

The monitoring and management tools provided by MongoDB Atlas are invaluable for database administrators and developers. It offers detailed performance metrics, real-time monitoring, and customizable alerts to help identify and resolve issues quickly. Automated backups and point-in-time recovery options provide data protection and disaster recovery capabilities.

Furthermore, MongoDB Atlas seamlessly integrates with other MongoDB tools and services. It works well with MongoDB Compass, a graphical user interface for MongoDB, and MongoDB Stitch, a serverless platform for building applications. The combination of these tools allows for a smooth development and deployment workflow.

MongoDB Atlas supports a wide range of cloud platforms, including AWS, Azure, and Google Cloud Platform, giving developers the flexibility to choose the provider that best suits their requirements. Additionally, Atlas offers multi-region deployments for improved data availability and disaster recovery.

In summary, MongoDB Atlas is a robust and user-friendly cloud database service that simplifies the deployment, management, and scaling of MongoDB databases. With its scalability, security features, monitoring tools, and integration capabilities, MongoDB Atlas provides a reliable foundation for building modern applications in the cloud.

**MongoDB:**

MongoDB is a popular NoSQL database that provides a flexible and scalable solution for storing and managing data. It differs from traditional relational databases by using a document-oriented approach, storing data in JSON-like documents with dynamic schemas. MongoDB offers several advantages, such as easy scalability, high performance, rich query language, support for replication and fault tolerance, and built-in geospatial and full-text search capabilities. With its flexible document model and powerful features, MongoDB is widely used in modern web and mobile applications, enabling developers to handle complex data structures and large volumes of data efficiently.



 Fig. 2 – ER diagram

**2.3. Express:**

Express. JS is the JavaScript library with best backend creation. The primary benefit of it is that it allows you to create web applications and not only because it is a kind of JavaScript library because means you can construct your application more easily and rapidly. Additionally, as we stated above, that JavaScript is the best backend programming language.

**Advantage of Using Express**

JS is that you’d be able to use JavaScript to do both the frontend and backend application. While there are many such platforms, they provide various language support for the frontend and backend, which makes interacting with them very difficult. But Express. JS never confronts any such problem.

**Pros**

The key benefit of using Express. JS is that you’d get quick application development experience with it.

Several such systems are unable to handle a higher level of requests but you will be able to manage requests effectively with the aid of Express. JS as it provides you with support for handling I / Q requests.

Express. JS is an open-source community.

Express helps you respond to requests with route support so that you may write responses to specific URLs

Supports multiple templating engines to simplify generating HTML.

**2.4. React:**

React is a powerful JavaScript library that revolutionized the way we build user interfaces for web applications. Developed and maintained by Facebook, React has gained immense popularity among developers due to its efficiency, flexibility, and reusability. With React, you can create dynamic and interactive UI components that seamlessly update in response to changes in data, providing users with a smooth and engaging experience.

One of the key principles of React is its component-based architecture, which allows you to break down your application into small, reusable pieces. Each component encapsulates its own logic and rendering, making it easier to build and maintain complex user interfaces. React also employs a virtual DOM (Document Object Model), which efficiently updates only the necessary parts of the UI when the underlying data changes, resulting in improved performance and responsiveness.

Furthermore, React promotes a declarative programming style, where you describe "what" you want your UI to look like, and React takes care of efficiently updating the actual DOM to match that desired state. This declarative approach simplifies the development process and reduces the potential for bugs.

React is not limited to the web browser; it can also be used to build native mobile applications using React Native. By leveraging the power of React, developers can write cross-platform applications that run smoothly on both iOS and Android devices, sharing code and speeding up development time.

Whether you are a seasoned developer or just starting your journey in web development, React offers a robust ecosystem with a vast collection of libraries, tools, and community support. Its popularity and extensive documentation make it easy to find resources and collaborate with other developers to enhance your React projects.

In summary, React empowers developers to build dynamic, reusable, and high-performance user interfaces for web and mobile applications. Its component-based architecture, virtual DOM, and declarative programming style make it a preferred choice for creating modern, interactive, and scalable applications.

**2.5. NodeJS:**

Node.js is a powerful and versatile JavaScript runtime environment that has revolutionized server-side development. It allows developers to build scalable and efficient network applications using JavaScript, which was traditionally limited to the client-side.

Node.js uses an event-driven, non-blocking I/O model, making it lightweight and highly performant. This enables handling a large number of concurrent connections and processing requests in an efficient manner. Whether you're building web servers, APIs, real-time applications, or command-line tools, Node.js provides a robust and scalable platform to meet your needs.

One of the key advantages of Node.js is its vast ecosystem of modules and packages available through the Node Package Manager (npm). This extensive collection of open-source libraries allows developers to quickly integrate functionality into their applications, saving time and effort.

Node.js is also well-suited for building microservices and implementing a modular architecture. Its module system enables developers to break down complex applications into smaller, reusable components, promoting code maintainability and reusability. Furthermore, Node.js has excellent support for asynchronous programming, making it easy to handle concurrent operations and avoid blocking the event loop.

With Node.js, you can leverage JavaScript's versatility and ubiquity to build both server-side and client-side applications. It provides seamless integration with frontend frameworks like React or Angular, enabling the development of full-stack JavaScript applications. Additionally, Node.js is cross-platform, running on various operating systems, including Windows, macOS, and Linux, making it accessible and adaptable to different environments.

Node.js enjoys a thriving and active community, constantly contributing new modules, tools, and resources. This vibrant ecosystem ensures that developers have access to the latest innovations and can find solutions to their challenges quickly.

In summary, Node.js empowers developers to build scalable, high-performance, and modular server-side applications using JavaScript. Its event-driven, non-blocking I/O model, extensive module ecosystem, and cross-platform support make it an ideal choice for developing a wide range of applications and services.

**CHAPTER 3 : SYSTEM ANALYSIS AND DESIGN**

**3.1. Use Case Diagram:**

Use-case diagrams illustrate and define the context and requirements of either an entire system or the important parts of the system. You can model a complex system with a single use-case diagram, or create many use-case diagrams to model the components of the system. You would typically develop use-case diagrams in the early phases of a project and refer to them throughout the development process.

Use-case diagrams are helpful in the following situations:

• Before starting a project, you can create use-case diagrams to model a business so that all participants in the project share an understanding of the workers, customers, and activities of the business.

• While gathering requirements, you can create use-case diagrams to capture the system requirements and to present to others what the system should do.

• During the analysis and design phases, you can use the use cases and actors from your use case diagrams to identify the classes that the system requires.

• During the testing phase, you can use use-case diagrams to identify tests for the system.

• Use cases represent only the functional requirements of a system. Other requirements such as business rules, quality of service requirements, and implementation constraints must be represented separately, again, with other UML diagrams.

****

Fig 3 – Use case Diagram

**3.2. Context Diagram:**

A [context diagram](https://miro.com/templates/context-diagram-template/) outlines how external entities interact with an internal software system.



Fig 4 – Context Diagram

It’s primarily used to help businesses wrap their heads around the scope of a system. As a result, they can figure out how best to design a new system and its requirements or how to improve an existing system.

Context diagrams are high-level diagrams, meaning they don’t go into the detailed ins and outs of the system. Instead, they map out an entire system in a way that’s simple, clear, and easy to understand.

For example, arrows are used to represent the flow of data between the system and each external element.

**3.3. Activity Flow Diagram:**

An activity diagram used to model a large activity’s sequential work flow by focusing on action sequences and respective action conditions. An activity diagram is represented by shapes that are connected by arrows. Arrows run from activity start to complete and represent the sequential order of performance activity. In this rectangle represents performances action, and described by text inside each rectangle. In this circle represents an initial workflow state or an end state. In this diamond shape is used shape is used to represents a decision, which is key activity diagram concept.

This activity flow provides a high-level overview of the user journey through the login, registration, event booking, ratings/reviews, and logout processes in the system. The actual implementation may involve additional steps and considerations based on the specific requirements of the system.



Fig. 5 - Activity Flow Diagram

* User is redirected to the registration page where they enter their details (name, email, password, etc.).
* After filling in the registration form, the user clicks on the "Register" button.
* The system validates the user's registration details and creates a new account.
* Once registered, the user is redirected back to the login page to log in with their newly created credentials.
* Upon successful login, the user is directed to the home page.
* In the home section, the user can view a list of upcoming events, featured events, or recommended events.
* The user can also navigate to the event section to browse through a comprehensive list of available events.
* In the event section, the user can view event details such as date, time, venue, description, etc.
* The user has the option to book an event by clicking on the "Book Now" button.
* The system validates the availability of tickets for the selected event and allows the user to proceed to the booking section.
* In the booking section, the user provides necessary details like the number of tickets and seating preferences (if applicable).
* The user completes the booking by clicking on the "Confirm Booking" button.
* The system processes the booking and generates a confirmation page with the booking details and a unique booking reference number.
* Optionally, the user can navigate to the ratings and reviews section to provide feedback on the event they attended.
* In the ratings and reviews section, the user can rate the event and leave comments or reviews about their experience.

**CHAPTER 4 : PROJECT PROFILE**

**4.1. Login page:**

****

Fig.6 – Login page

User Authentication: The login page serves as the entry point for users to access the application. When users submit their credentials, the server-side code verifies the validity of the provided information against the user database.

JWT Generation: Upon successful authentication, the server generates a JWT token. This token contains encrypted user information, such as the user's ID or username, along with an expiration date. The JWT token is then sent back to the client as a response.

Client-side Storage: The client-side code stores the received JWT token securely, typically in local storage or cookies. This token is crucial for subsequent requests made by the authenticated user.

Authorization Middleware: To protect routes or API endpoints, a middleware function can be implemented in the server-side code. This middleware verifies the validity of the JWT token sent by the client with each protected request. If the token is valid and has not expired, the request is allowed to proceed; otherwise, the user is redirected to the login page.

**4.2. Register page:**



Fig. 7 – Register page

User Registration Form: The register page provides a form where users can input their registration details, such as username, email address, password, and any additional required information. Client-side validation can be implemented to ensure data integrity and to provide real-time feedback to users.

Password Encryption: When users submit their registration details, it is crucial to securely encrypt their passwords. The server-side code should incorporate strong encryption algorithms, using bcrypt, to store password hashes in the database instead of plain text passwords.

User Database: The server-side code should include functionality to store user registration details in a secure and reliable database, such as MongoDB. User information such as username, email, and hashed passwords should be stored for future authentication and authorization processes.

JWT Generation: After successful registration, the server generates a JWT token for the newly registered user. This token includes encrypted user information and can be used for subsequent authentication and authorization.

Automatic Login: Upon successful registration and JWT generation, the server can automatically log in the user by sending the JWT token back to the client-side code. The client-side code should securely store the token for future requests.

**4.3. Booking Section:**

Event Listing: Display a list of available events, including relevant details such as event name, date, location, and availability. This allows users to browse and select the events they are interested in booking.

Event Details: Provide comprehensive information about each event, including a detailed description, schedule, ticket types, pricing, and any special instructions. Clear and visually appealing event pages help users make informed decisions.

User Authentication: Implement a secure user authentication system to ensure that only authenticated users can make bookings.



Fig.8 – Booking Section

**4.4. Review Section:** their experience, highlights, or suggestions.

Rating System: Implement a rating system, such as a star rating or a numerical scale, to allow users to rate different aspects of the event, such as organization, venue, speakers, or overall satisfaction. This provides a standardized way for users to express their opinions.

Review Display: Display the reviews in an organized and easily readable format. Include the reviewer's name, rating, review content, and the date of submission. Consider incorporating pagination or infinite scrolling for better user experience, especially if there are a large number of reviews.

Sorting and Filtering: Provide options for users to sort reviews based on relevance, rating, or date. Additionally, consider implementing filters to allow users to view reviews by specific criteria, such as the type of event or location.



Fig.9 – Review Section

**4.5. Database:**

Documents: A collection consists of individual JSON-like documents, which are the basic unit of data storage in MongoDB. Each document represents a single record and can contain multiple fields with their corresponding values. These documents are stored in BSON (Binary JSON) format.

Indexing: Collections can be indexed to improve query performance. Indexes are data structures that store the values of specific fields, allowing for efficient searching and sorting of documents. MongoDB supports various types of indexes, including single field indexes, compound indexes, text indexes, and geospatial indexes.

CRUD Operations: Collections support the standard CRUD (Create, Read, Update, Delete) operations. Data can be inserted into a collection using the "insert" operation, read using "find" or "findOne" operations, updated using "update" operations, and deleted using "remove" operations.



Fig.10 – MongoDB Database