

**EventEase: Ticket Booking Management System**

**Worcester Polytechnic Institute**

**CS 542 – Database Management Systems**

Anushka Bangal, Christina Berthiaume, Ayush Shinde

May 3, 2023

## **Abstract**

Our project aims to implement a ticket booking management system that enables users to book tickets for various events, such as movies, concerts, and sports events. The system also allows users to view the details of upcoming events, including their performers, venues, ticket prices, etc. In order to do this, we design and develop a database management system and integrate it with a front-end interface. The database stores all the data related to events, venues, customers, and orders in a relational database using SQL, which enables efficient management and retrieval of information. The front-end interface allows users to explore different events in their area and easily book tickets to multiple shows.

To achieve our objective, we followed a systematic approach, starting with drawing the Entity Relationship Diagram (ERD) and writing down the schemas for the database management system. After that, we began working on the front-end interface. We designed the front-end interface using HTML, JavaScript, and CSS. Finally, we connected the front-end to our database using PHP and JavaScript.

**Keywords:** Database management, Entity-relationship model, ticket booking, user, SQL, JavaScript, HTML, CSS, API, user interface design.

## **Overview**

In the era of online ticket booking, it is difficult to keep track of all the different ticket booking websites. Purchasing tickets for respective events can be time-consuming and confusing, especially when there are multiple events happening at the same time. Many of these events may have their own platform for selling tickets. For instance, sports fans often have to buy tickets from different platforms for different sporting events such as football, baseball, and basketball. Further, events may not be well advertised, leading customers to miss out.

We wanted to create a database application whose purpose is to allow users to book tickets for various events all on one platform. After creating an account, users will be able to browse different events going on in their area and see information about the event, including feedback from other users. We want our application to be interactive, allowing users to communicate with each other and share their thoughts about a particular event. Once the user determines what event they want to attend, they choose their seat and check out. The user enters their payment information into our payment gateway, and if the payment is successful the ticket is added to their account. EventEase allows users to buy multiple tickets for multiple shows simultaneously. Our system aims to be efficient, user-friendly, and easy to use for both the users and the administrators.

The primary tools that we will use are MySQL for the backend, HTML, JavaScript and CSS for the front-end, and a combination of PHP and Xampp to ensure that different components of the application work together. HTML JavaScript and CSS make up the framework of the website, and PHP sends the data from the website to be stored in our SQL tables. Our project aims to provide a centralized platform that makes it easier for users to book tickets for a variety of events.

## Background Material

Before beginning our project, we researched other projects and resources that would be helpful. Below is the list of projects and other references that we have utilized for our project:

- <https://github.com/topics/ticket-booking-system>

This reference includes ten public repositories for ticket booking systems. Each one is coded in different programming languages and includes different features. We will reference these when designing our project to decide which languages will be the best for our project, and what features we should include.

- [https://github.com/ImDP/Movie\\_Tickets](https://github.com/ImDP/Movie_Tickets)

This is a movie ticket booking system that uses the same programs we plan to use, JavaScript/HTML/CSS, thus we thought it would be good to reference for the front-end development.

- “MOVIE TICKET BOOKING MANAGEMENT SYSTEM”  
([https://www.academia.edu/31536019/MOVIE\\_TICKET\\_BOOKING\\_MANAGEMENT\\_SYSTEM\\_PROJECT\\_REPORT\\_doc](https://www.academia.edu/31536019/MOVIE_TICKET_BOOKING_MANAGEMENT_SYSTEM_PROJECT_REPORT_doc))

This project is very similar to ours. It is a movie ticket booking management system. The paper includes many things that can be helpful to our project, such as ERDs, database design, and performance testing. This will be helpful while we design and test our database.

- “Database Management Systems” - Book by Johannes Gehrke and Raghu Ramakrishnan.

We plan to reference the textbook for any technical questions we have.

- <https://github.com/Akibsheikh48/movie-seat-booking>

This project is a movie ticket booking system that allows the user to pick their seat from a diagram showing seats that are free. Since we plan on designing a similar seat choice system, we thought this project would be a good reference for how to do that.

- “BookMyShow” website: <https://in.bookmyshow.com/explore/home/pune>

This website is very similar to our project idea, and therefore we plan to base our project on it. We would like to try to create a similar-looking website. We will also try to recreate some features of the website in our own project, such as the sliding advertisements shown at the top of the page.

- “Ticket Master” website: <https://www.ticketmaster.com/>

Our project idea closely resembles this website, so we intend to use it as a model for our project. In our project, we plan to incorporate a search bar with location functionality, similar to the one present on this website.

- <https://readthedocs.org/projects/event-reservation-system/downloads/pdf/latest/>

This is a research paper we found helpful in event reservations. It also emphasizes different events being conducted and allows users to book any event.

- <https://github.com/nikhilpatil99/Stadium-Seat-Booking-System>

This open-source project helped us build the user interface for the stadium seat booking.

- “A PROJECT ON ONLINE TICKET BOOKING SYSTEM” by Punyaslok Sarkar, Mrs. Sherly Noel  
([https://www.researchgate.net/publication/342466860\\_A\\_PROJECT\\_ON\\_ONLINE\\_TICKET\\_BOOKING\\_SYSTEM](https://www.researchgate.net/publication/342466860_A_PROJECT_ON_ONLINE_TICKET_BOOKING_SYSTEM)).

This research shows the implementation of PHP and SQL for front-end and database design respectively. It also discusses the user authentication, seat selection, and payment gateway integration techniques which would act as a valuable resource to refer to for our project.

From these references, we were able to gather many insights about our project. The first step in creating a database management system is determining the physical design of our application. The physical design includes terms on how the data is input into the system, how it is verified/authenticated, how it is processed, and how it is displayed as output. It is the input and output processes of the system. Therefore, how we design our database is crucial to the usability of our application. Another important aspect of building a database management application is testing. We must test our application in order to identify errors in the system, such as incorrect functions, and make sure that it is complete and accurate. We must also ensure that all of the components work together effectively as a single system. Finally, in order to implement the website, we took inspiration from many of these projects. It helped us determine which methods would work best for developing the user interface. We modeled our website after the *BookMyShow* website. We used this website to help us with the layout of the user interface.

## Approach

Our project began with requirement analysis and planning, where we identified the necessary features and functionalities of the ticket booking management system. We gathered information about the events the system would handle, collected details about potential users, including customers and event organizers, and determined the data that needed to be stored and managed in the database. During this phase, we also planned the project timeline, assigned tasks to team members, and set milestones for each project phase.

In the database design phase, we developed an Entity-Relationship Diagram (ERD) to visualize the backend structure of the database application. We then transformed the ER diagram into a logical model suitable for designing a relational database management system (RDBMS). We normalized the database schema to reduce redundancy and improve data integrity. Finally, we defined primary keys, foreign keys, partial keys, and other constraints to optimize database performance.

Next, we focused on backend development, which included creating the database schema in SQL, establishing tables, and implementing relationships between them. We wrote queries for selecting, inserting, updating, and deleting data in the database. In order to optimize database performance, we indexed frequently queried data.

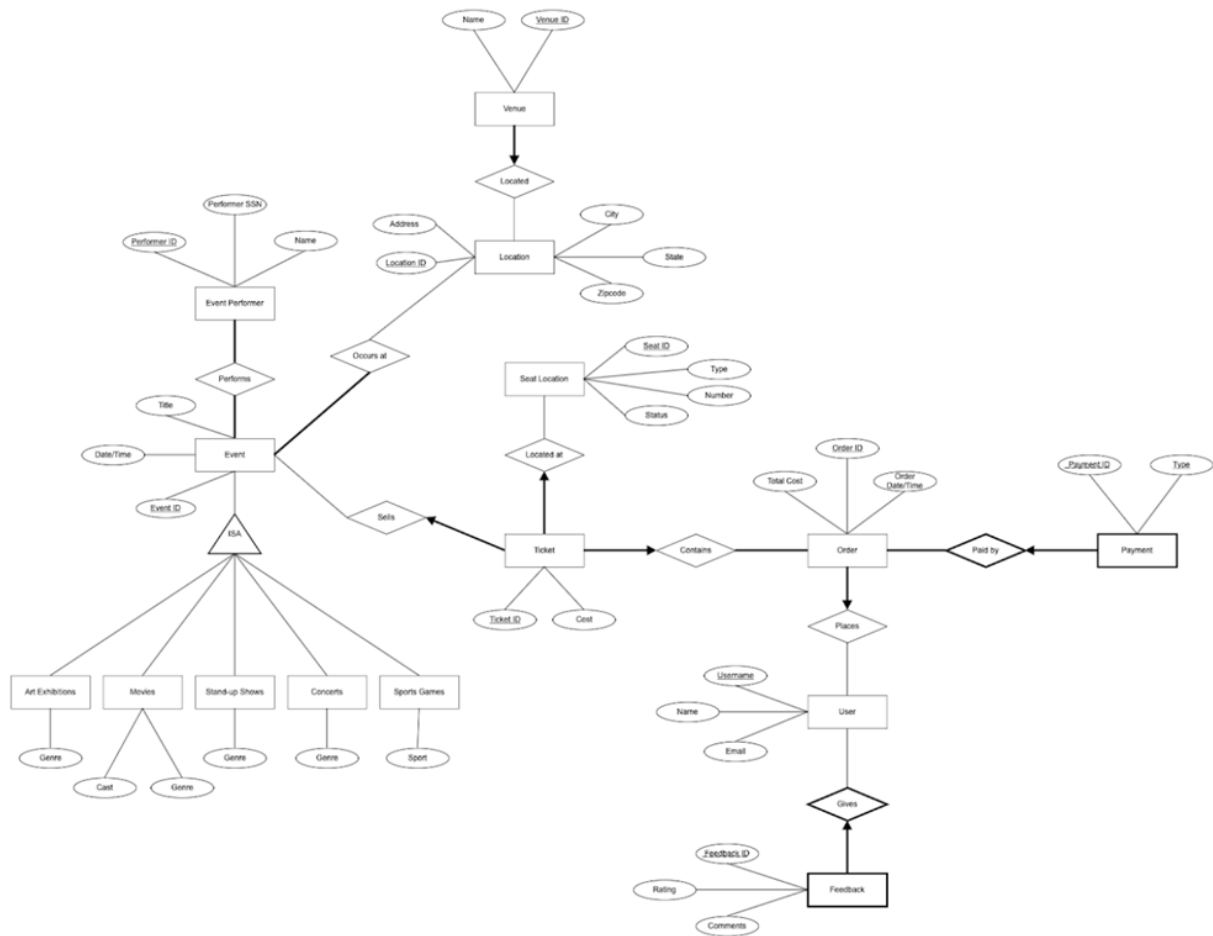
During the front-end development phase, we designed a user interface for the database application using JavaScript, HTML, CSS, and PHP ensuring a clean and user-friendly experience. We incorporated interactive features, such as event search, user registration, and ticket booking using JavaScript. We employed responsive design techniques to guarantee compatibility with different devices and screen sizes.

After developing the front-end and back-end, we integrated them using PHP. We conducted integration tests to ensure different components of the application worked together correctly. We performed extensive testing, including functionality, usability, compatibility, and performance testing to provide a seamless user experience across various browsers and devices.

The final step in our project was deploying the website on a web server. We chose a suitable hosting provider, set up the web server, and configured the database server settings for optimal performance. We continually monitored the application for errors and performance issues, adjusting as needed to ensure a stable and responsive platform.

Throughout the project, we documented our design choices, code implementations, and testing procedures. We created user manuals and tutorials to help users understand the system's features and functionalities. Additionally, we established a support system to address user concerns and issues, ensuring continuous improvement of the application based on user feedback.

Below is a rough outlook of the project in the form of an ERD diagram:



Based on the ER diagram above, we created relational schemas for each of the entities and relationships. These schemas were then transformed into tables using MySQL.

## Test Cases

ID	Test Cases	Input Value	Expected Output	Pass/Fail	Date Tested
1	Test Procedure: Login with username and password.	User: Kate Miller  Password: kate123	Successful login.	Passed	4/16/23
	Username: Kate Miller  Password: kate123	User: Kate Miller  Password: katemiller	Username/Password doesn't match.	Passed	4/21/23
	Expected Results: Successful login. Otherwise, an error message.	User: Chandler Bing  Password: chandler123	Username/Password match.		

2	<p>Test Procedure: Check for available seats at an event.</p> <p>Expected Result: Depending on whether there are tickets left, it will show the seats that are still available. If none are left, an error message will be shown.</p>	<p>Event Name/Performer Number of Tickets (tickets available)</p> <p>Event Name/Performer Number of Tickets (sold out)</p>	<p>Shows seats that are still available for that show and their price.</p> <p>Sorry, this event is sold out.</p>	Passed	4/19/23
3	<p>Test Procedure: Select a seat.</p> <p>Expected Result: If the seat is available, it will add the ticket to your cart. If the seat is not available, it will show an error.</p>	<p>Seat number (Available)</p> <p>Seat number (Taken)</p>	<p>Add a ticket.</p> <p>Sorry, this seat is already taken.</p>	<p>Passed</p> <p>Passed: Does not let the user to select that seat. (for movies )</p>	4/20/23



4	<p>Test Procedure: Select multiple seats.</p> <p>Expected Result: User can select one or multiple seats that are still available. Raises error if user tries to purchase more than five tickets.</p>	<p>Seat number (one seat)</p> <p>Seat numbers (<math>n \leq 5</math> seats)</p> <p>Seat numbers (<math>n &gt; 5</math> seats)</p>	<p>Adds one ticket to cart.</p> <p>Adds <math>n</math> tickets to cart.</p> <p>Sorry, there is a limit of five tickets per user.</p>	TBD	TBD
5	<p>Test Procedure: Submit payment information.</p> <p>Expected Result: If the transaction goes through, success! Tickets are sent to the user's email. If the transaction fails, error message.</p>	<p>Credit/Debit card information (sufficient funds)</p> <p>Credit/Debit card information (insufficient funds)</p> <p>Credit/Debit card information (incorrect information)</p>	<p>Congrats! You are going to see <i>[insert event performer]</i>. Tickets will be sent to your email.</p> <p>Card declined. Please try again.</p> <p>Credit/Debit card information invalid. Please try again.</p>	TBD	TBD

6	<p>Test Procedure: Insert new event.</p> <p>Expected Result: If the venue is available for the selected date/time, the event is added to the database and tickets for the event become available. If the venue is not available, error message.</p>	<p>Event ID, title, type, performer ID, location, date/time (venue available)</p> <p>Event ID, title, type, performer ID, location, date/time (venue unavailable)</p>	<p>Success! Your event <i>[insert event title]</i> is booked for <i>[insert date/time]</i> at <i>[insert venue name]</i>! Adds an event to database.</p> <p>Sorry, this venue has already been booked for <i>[insert date/time]</i>.</p>	Passed	4/28/23
7	<p>Test Procedure: Update event.</p> <p>Expected Result: Updates specified attribute of the event to a new value. If the attribute that the user is trying to update is the primary key, error message.</p>	<p>Event ID, attribute to update, new value (attribute is not the primary key)</p> <p>Event ID, attribute to update, new value (attribute is the primary key)</p>	<p>Success! Event <i>[insert event ID]</i> has been updated. Updates event attribute to new value.</p> <p>Sorry! This attribute cannot be changed.</p>	Passed	4/27/23

8	<p>Test Procedure: Delete event.</p> <p>Expected Result: Removes the event from the database, along with all of its attributes and foreign keys.</p>	Event ID	Event is deleted from database. (in PHP Admin)	Passed	4/25/23
9	<p>Test Procedure: Select event, performer, venue, etc.</p> <p>Expected Result: Selects specified event, performer, or venue and shows all available tickets for that event. If there is no such event, performer, or venue, error message.</p>	<p>Event ID, performer ID, or venue ID (event exists)</p> <p>Event ID, performer ID, or venue ID (event doesn't exist)</p>	<p>Shows table of the event, performer, or venue selected and its available tickets.</p> <p>Sorry! The event, performer, or venue you are looking for doesn't exist.</p>	TBD	TBD

Our team faced significant challenges during the implementation of the front-end and its integration with the back-end, resulting in incomplete test case validation. Nevertheless, these test cases served as a clear representation of our application's intended functionalities and features. Although we were able to execute SQL queries such as "SELECT," "INSERT," "DELETE," and "UPDATE" to

manipulate data from the back-end, we encountered difficulties in integrating these functionalities with the front-end of the application. While we achieved the successful implementation of the seat selection feature, we faced challenges in linking it with other aspects of the application.

## **Lessons Learned**

Throughout the course of this project, we have applied various concepts acquired in class to real-life scenarios, making critical decisions that a database designer would encounter. Our decisions regarding which entities were significant to the project, their relationships, attributes, primary keys, and other crucial aspects have had a significant impact on the application's final product. Even a small alteration in the Entity Relationship Diagram (ERD) could have a considerable effect on the database.

One example of such an impactful decision was determining whether payment should be an attribute of the Order entity or its own entity. Initially, we considered it to be an attribute, but upon further reflection, we concluded that it would be more appropriate as a weak entity dependent on the Order entity. This decision resulted in a significant alteration to our database.

After finalizing the ERD, we developed relational schemas, identifying candidate keys and foreign keys. We carefully considered the project's real-life context while making these decisions. We had to examine the unique attributes of each entity and identify weak entities and foreign key constraints. For instance, we initially classified Datetime as a weak entity in our first ERD draft but later corrected it to a strong entity since it doesn't rely on anything. This experience taught us the importance of accurately identifying these keys.

As we progressed to the backend development phase, we encountered new challenges and gained valuable insights. With the ERD and relational schemas complete, we created our tables in SQL and wrote queries for selecting, inserting, updating, and deleting data. We faced query performance issues and learned the significance of optimizing database performance by indexing frequently queried data. This helped to minimize response times and enhance the overall user experience.

Moving to the front-end development phase, we were faced with a new set of decisions and questions. One of these questions was the number of events to include in our project. We also implemented a date picker to allow users to select their preferred date and time, and the database would display available tickets for events during that period.

The front-end development phase exposed us to the intricacies of designing a user-friendly interface using JavaScript, HTML, and CSS. We carefully considered user needs and preferences, ensuring our application was visually appealing and easy to navigate. We created test cases to verify the correctness and performance of the user interface and added interactive features using JavaScript to enhance the user experience.

During the integration phase, we were challenged with ensuring seamless communication between the front-end and back-end components. We wrote integration tests to ensure that all components worked together correctly and tested the application across various browsers and devices to guarantee compatibility. The deployment phase emphasized the significance of monitoring the application for errors and performance issues. Configuring the web server and database server settings for optimal performance was critical in ensuring a smooth user experience.

Throughout the project, we continued to learn valuable lessons and adjusted our database accordingly. The experience highlighted the importance of carefully considering each decision, understanding the real-life context of the project, and remaining adaptable to changes. Effective communication and collaboration within the team were also recognized as essential, as well as the need for adaptability and flexibility in the face of unexpected challenges. As we move forward, we intend to apply these lessons to future projects, becoming more efficient and effective in our database design and implementation. In conclusion, this project has been an enriching learning experience, providing us with a deeper understanding of database design, frontend and backend development, integration, and deployment.

## **Conclusion**

Our ticket booking management system project has undergone a comprehensive development process that began with drafting the Entity Relationship Diagram (ERD) and resulted in the successful completion of both frontend and backend development. Throughout this journey, we have gained invaluable insights and practical experience in various aspects of database design, SQL, and user interface design.

Our initial phase involved the creation of the ERD, where we made crucial decisions regarding entities, relationships, and attributes. We carefully refined the diagram by identifying weak entities, fixing constraint notations, and adjusting primary keys to ensure a solid foundation for our database. The next step was to transform the ERD into relational schemas, where we defined primary keys, candidate keys, and foreign keys for each schema. This allowed us to create our tables in SQL and test the functionality of several SQL statements, such as "INSERT", "UPDATE", "DELETE", and "SELECT".

The backend development phase focused on creating the database schema in SQL, implementing relationships between tables, and optimizing the database's performance. With the backend development complete, we proceeded to the frontend development phase.

Frontend development involved creating an attractive and user-friendly interface using JavaScript, HTML, CSS, and PHP. We designed and implemented essential features such as user login, registration, ticket booking, and profile management. We also ensured that our user interface was compatible with various browsers and devices, offering a seamless experience to end users. Finally, we integrated the front-end and backend components to create a cohesive and efficient ticket booking management system.

Throughout this project, we have encountered numerous challenges. The main challenge for us was integrating all the pieces of the front-end and back end. We struggled in deciding which methods would work best for our database. We found the technical aspects of connecting the front-end with the back end to be very challenging. While we were able to connect pieces to our database, we were unable to connect it all. There were multiple features that we would've like to implement into our database if time had allowed.

In summary, the entire process from ERD drafting to front-end and back-end development has provided us with a wealth of knowledge and experience in database management and application development. Our team has successfully designed and implemented a functional ticket booking management system, covering various aspects such as drafting the ERD, writing relational schemas, and completing both front-end and back-end development. However, there were certain areas where we fell short, such as writing stored procedures and triggers for data manipulation. These components would have further optimized the system's performance and ensured better data integrity. However, we believe this is a good prototype for what we were trying to do. In the future, we can continue to add to it and improve the application. We are confident that our ticket booking management system will serve as an effective and reliable tool for users and event organizers alike.

Task	Description
User Registration	Allow users to create an account by providing their personal information like name, email, phone number, etc.
User Login	Allow registered users to log in with their email/username and password to access the system.
Event Management	Allows the system administrator to manage events, including adding new events, modifying event details (such as event name, date, time, location, and ticket prices), and removing events that have already passed.
Ticket Management	Allows the system administrator to manage tickets, including adding new tickets, modifying ticket details (such as ticket price, availability, and seat location), and removing tickets that have already been sold out.

Ticket Booking	Allows users to search for events and book tickets online by selecting the event they are interested in, the number of tickets they want to purchase, and the payment method.
Payment Processing	Allows users to pay for their tickets using various payment methods like credit/debit cards, PayPal, or other online payment systems.
Order Management	Allows users to view their orders and cancel their orders if necessary.
User Profile Management	Allows users to view and update their personal information, manage their saved payment methods, and view their order history.

## Future Work

Our application has some unimplemented features due to technical challenges faced during the development phase. To enhance user engagement, we intended to add interactive features, such as allowing users to provide feedback and make comments on events. The feedback feature would serve as a resource for other users to make informed decisions on attending specific events. Additionally, we planned to provide seating maps for all venues, allowing users to choose their seats. Although this feature was implemented for movies, we aspired to expand it to all events.

Given more time, there are numerous areas in which we would like to improve the application. One way is to integrate third-party payment gateways such as PayPal or Apple Pay to simplify transactions, instill trust, and provide additional convenience to users. We also envisioned integrating social media platforms into our application to increase visibility, promote events, and attract a wider audience. Developing a mobile application for both iOS and Android would enhance accessibility and the user's experience by enabling customers to book tickets and manage their profiles while on the go.

To ensure stable and efficient operations as the system grows, we would conduct research on optimizing the database for large-scale operations and high-traffic loads. Surveys would also be conducted to ascertain user preferences, likes, and dislikes to improve user experience. Additionally, we would integrate a recommendation engine that suggests events based on user preferences, booking history, or similar interests to enhance user engagement and promote ticket sales. We would also develop a comprehensive analytics module that generates insights and reports for event organizers and system administrators to make data-driven decisions to improve events and overall user experience. Finally, developing an integrated customer support module that handles user inquiries and issues would improve the overall user experience and foster customer loyalty.

As the ticket booking management system continues to grow and evolve, it is crucial to identify potential areas for future research and development to ensure the system remains relevant, practical, and user-friendly. By exploring these opportunities, we can create a more robust and efficient application that caters to the diverse needs of users and event organizers alike.

## Appendix

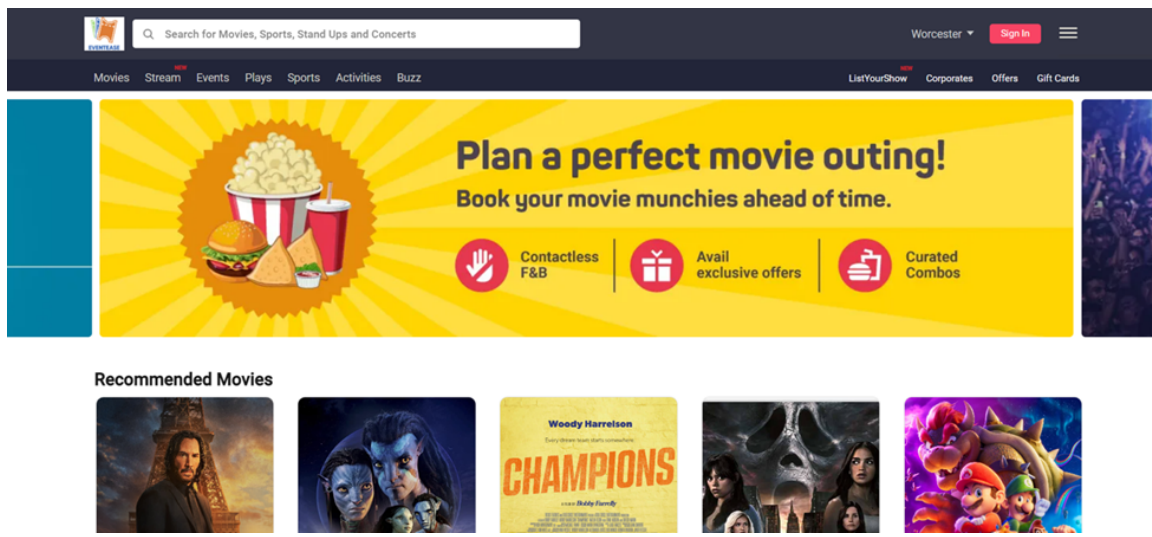


Figure 1: Home Page

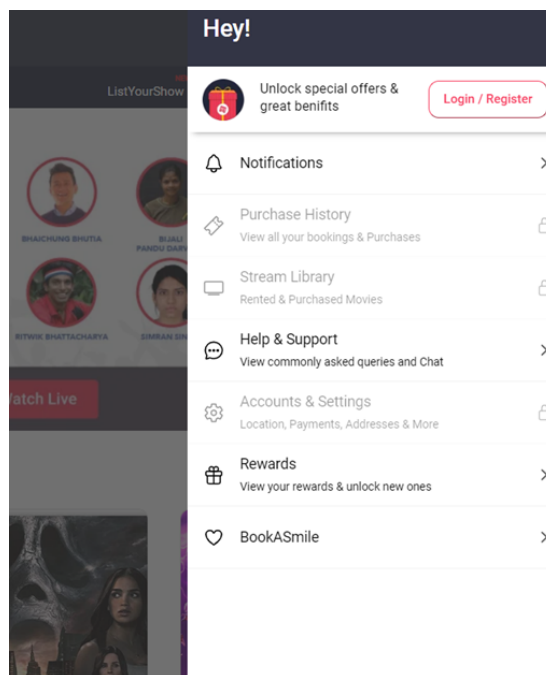


Figure 2: Side Bar



Log InRegister

Username

Password

☐ Remember Password

Log In

Log InRegister

Name

Username

Password

Email

Register

Figure 3: Login and Register Page

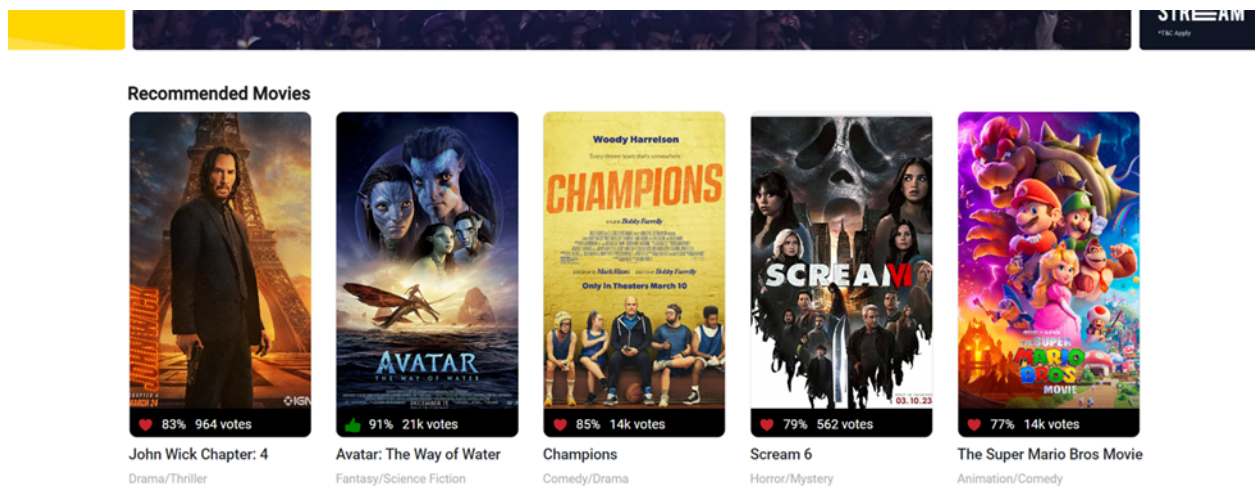


Figure 4: Movies

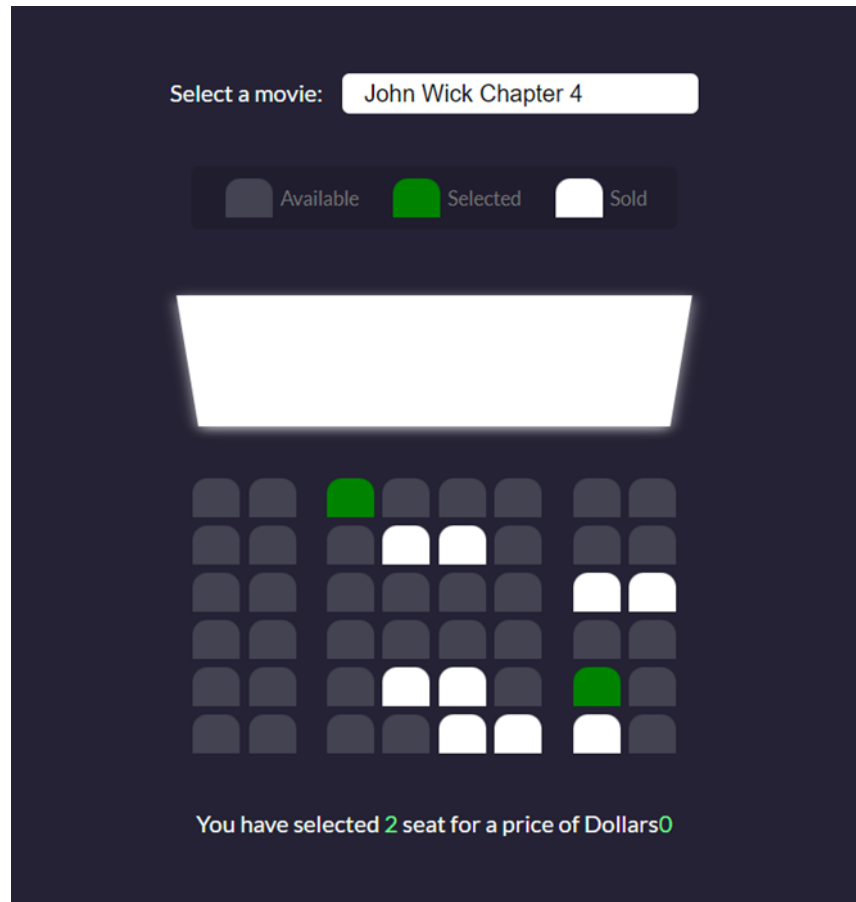


Figure 5: Seat Selection Page

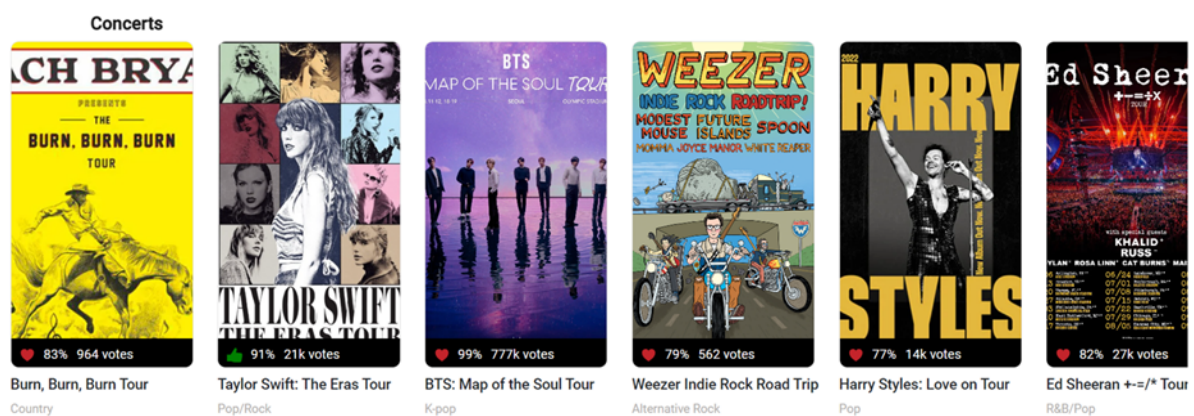


Figure 6: Concerts

## Sports

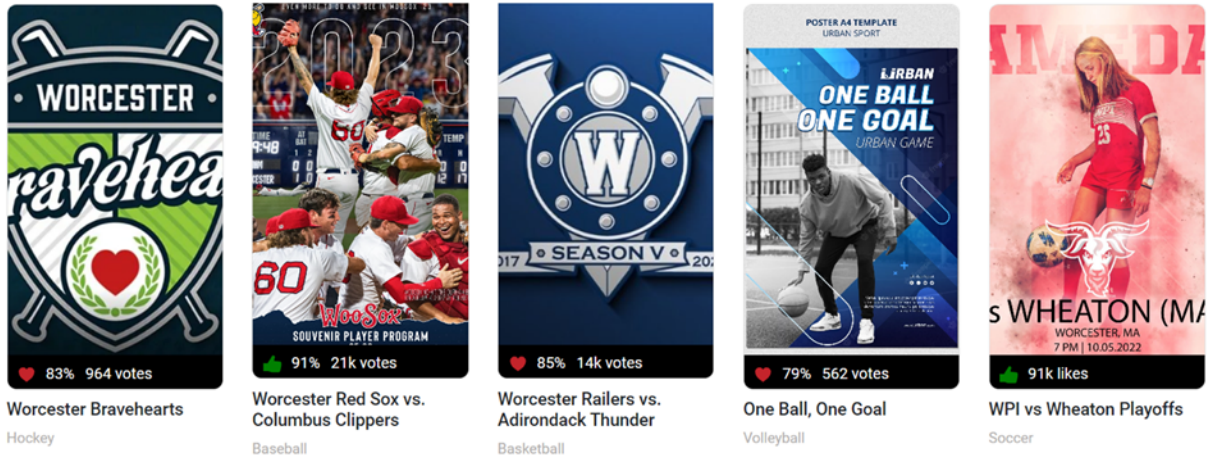


Figure 7: Sports

EventEase

Book Ticket for

Price: \$

Full Name:

Email:

Phone Number:

Number of Tickets:

Book Ticket

Figure 8: Ticket Booking Page





BILLING ADDRESS		PAYMENT	
Full Name :	<input type="text" value="John Deo"/>	Accepted Cards :	   
Email :	<input type="text" value="Example@Example.Com"/>	Name On Card :	<input type="text" value="Mr. John Deo"/>
Address :	<input type="text" value="Room - Street - Locality"/>	Credit Card Number :	<input type="text" value="1111-2222-3333-4444"/>
City :	<input type="text" value="Mumbai"/>	Exp Month :	<input type="text" value="January"/>
State :	<input type="text" value="India"/>	Zip Code :	<input type="text" value="123 456"/>
		Exp Year :	<input type="text" value="2022"/>
		CVV :	<input type="text" value="1234"/>
<input type="button" value="Proceed To Checkout"/>			

Figure 9: Payment Gateway