CS 5200 Project Report

Team name: GuptaVUmrajkarGKharcheC
Team members: Vandit Gupta, Gargi Umrajkar, Chaitanya Kharche

Project Description

EventBuzz is a simple event management application designed to streamline the planning, organization, and attendance of events, whether they are small community gatherings, corporate conferences, music festivals, or sports events. With a user-friendly interface, EventBuzz empowers event organizers, attendees, and sponsors to create, discover, and engage with events like never before.

Core Functionalities
CRUD Operations:
At the heart of EventBuzz is its robust capacity to perform Create, Read, Update, and Delete (CRUD) operations, providing users with complete control and flexibility over event management and interaction:
Create: Users can effortlessly create new events, adding comprehensive details such as event categories, dates, locations, and ticketing options. Users can also add reviews. Organizers and sponsors can add their information.
Read: EventBuzz offers an intuitive and user-friendly platform for attendees to browse and discover events. Users can view detailed information about events, organizers, venues, and ticketing options, making it easier to find events that align with their interests.
Update: The application allows organizers to update event details as needed, ensuring that attendees have access to the most current information. This includes changes in dates, venues, ticket availability, and more.
Delete: Organizers have the flexibility to delete events when necessary. This feature is critical for maintaining an up-to-date and accurate listing of events.

Key Highlights

- **Effortless Event Creation**: Organizers can create, configure, and customize event listings with intricate details, dates, locations, and ticket pricing, all facilitated through an intuitive interface.
- **Derived Attribute Display**: EventBuzz skillfully utilizes derived attributes in its data tables, enabling dynamic and real-time presentation of calculated data. This feature enhances the user's ability to view comprehensive event details, like the total sponsorship amount by a sponsor, etc.
- **Procedures for Data Management**: The application employs sophisticated stored procedures in the backend, streamlining complex CRUD operations.
- **User Event Tracking:** The platform showcases the number of events a user is registered for using User Defined Functions rendered from SQL, providing them with a clear and accessible record of their upcoming activities and engagements
- **Registrant Count**: A user-defined SQL function is implemented to calculate and showcase the number of users registered for each event, offering a clear view of participant engagement and event popularity.
- **Trigger-Based Data Updates**: EventBuzz leverages database triggers to automatically update certain values in response to specific data changes. For instance, when a new ticket is booked or an event's detail is modified, relevant changes are immediately reflected across the platform, maintaining data integrity and consistency.
- **Efficient Error Handling:** EventBuzz implements robust error-handling mechanisms on both client and server sides. Forms are meticulously designed to accept only correct data types, promoting clean and accurate data entry.

- **User and Error Logs:** To maintain a high level of operational integrity, the system includes user logs and error logs tables. These log tables are crucial for tracking successful and faulty requests, aiding in quick resolution of issues and continuous monitoring of system performance.

Data Description

In the Event Buzz application, a comprehensive array of interconnected tables forms the foundation of this dynamic event management system. The "Users" table records user information, while the "Organizers" table categorizes organizers with details about their organizations. The "Venues" table manages venue information, and the "Events" table captures event specifics, connecting them to respective organizers and venues. Ticket-related data is stored in the "Tickets" table, while "Orders" tracks user orders for event tickets, and "Reviews" provides user feedback on events. Notifications are stored in the "Notifications" table, and "Sponsorships" track financial support for events. These tables create an intricate network to facilitate user interactions, event management, and data management in Event Buzz.

Domain Interest

Creating an event management application is driven by three core motivations. It's a canvas for creativity, a challenge for quick thinkers, and a platform for fostering connections. This exciting journey allows us to blend artistry, problem-solving, and the joy of bringing people together.

README

Technical Specifications

Backend:

- Language: JavaScript
- Framework: Node.js along with Express.js

Frontend:

- Language: JavaScript
- Framework: React.js

Database:

- MySQL

Getting Started - Prerequisites

Before you begin, ensure you have the following installed:
Node.js: Download Node.js
MySQL: Download MySQL
MySQL Workbench: Download MySQL_Workbench

Installation - Setup

Backend Setup - Navigate to the backend directory on the terminal:
Run on the terminal: “cd eventbuzz_backend”
Install dependencies:
Run on the terminal: “npm install”

Frontend Setup - Navigate to the frontend directory on the terminal:
Run on the terminal: cd eventbuzz_frontend
Install dependencies:
Run on the terminal: “npm install”

Database Setup
Run the provided SQL dump

OR

Run the SQL scripts provided in the eventbuzz_database folder in the following order:

1. CreateSchema.sql - This file creates the base tables
2. CreateAuditTables.sql - This file creates the log tables
3. Indexes.sql - This file creates indexes for the base tables and the log tables for improved performance
4. StoredProcedures.sql - This file creates the Stored Procedures for CRUD operations on all table
5. Triggers.sql - This file creates the Triggers for UserLog and ErrorLog tables
6. UDFs.sql - This file creates the UDFs used in the project
7. InsertRecords.sql - This file inserts 10 records each in all the base tables

Running the Application

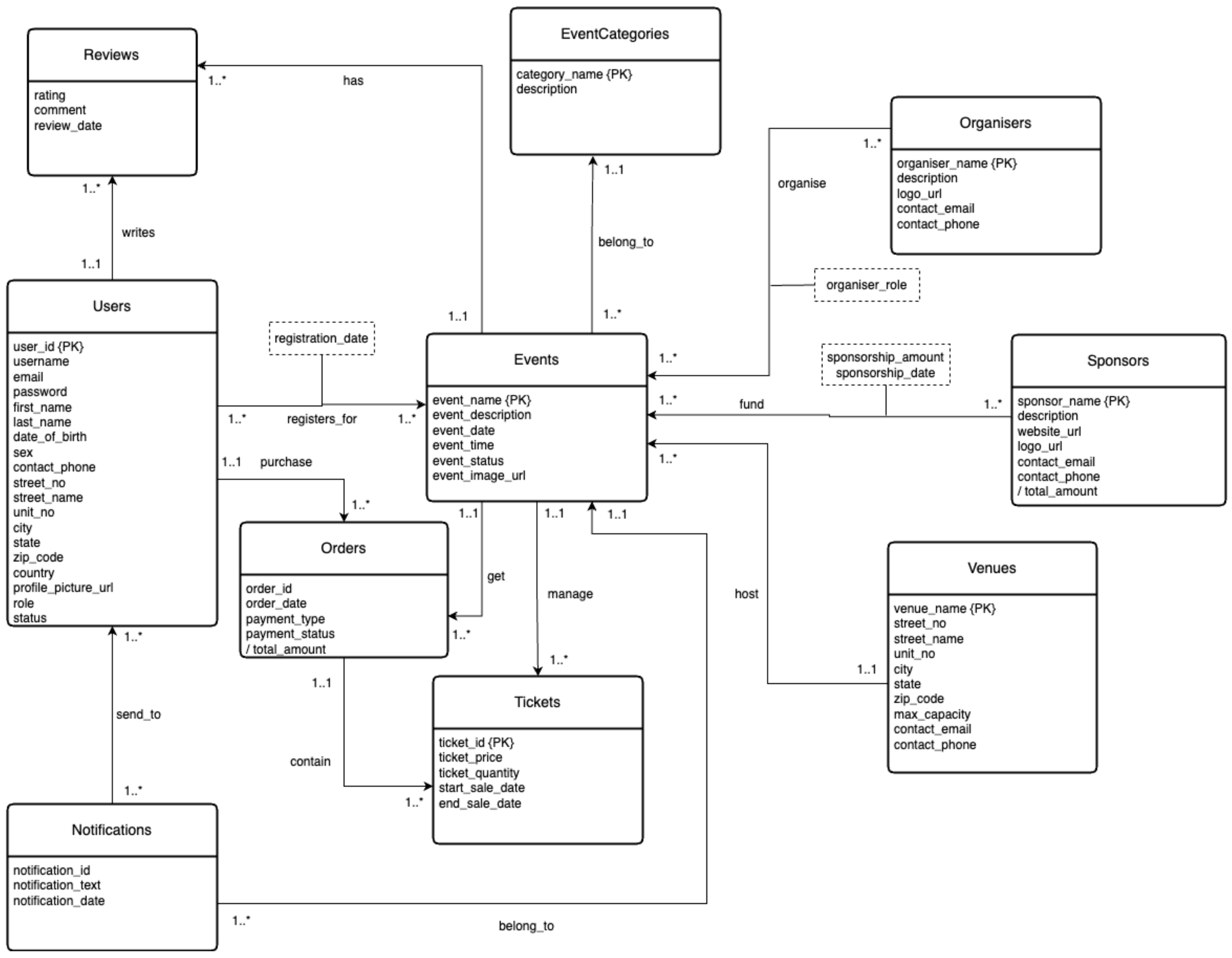
Open a terminal and navigate to the folder eventbuzz_backend
Run on the terminal: “npm start”

NOTE - Change the ‘`DB_PASSWORD`’ in the app.js file to the localhost password

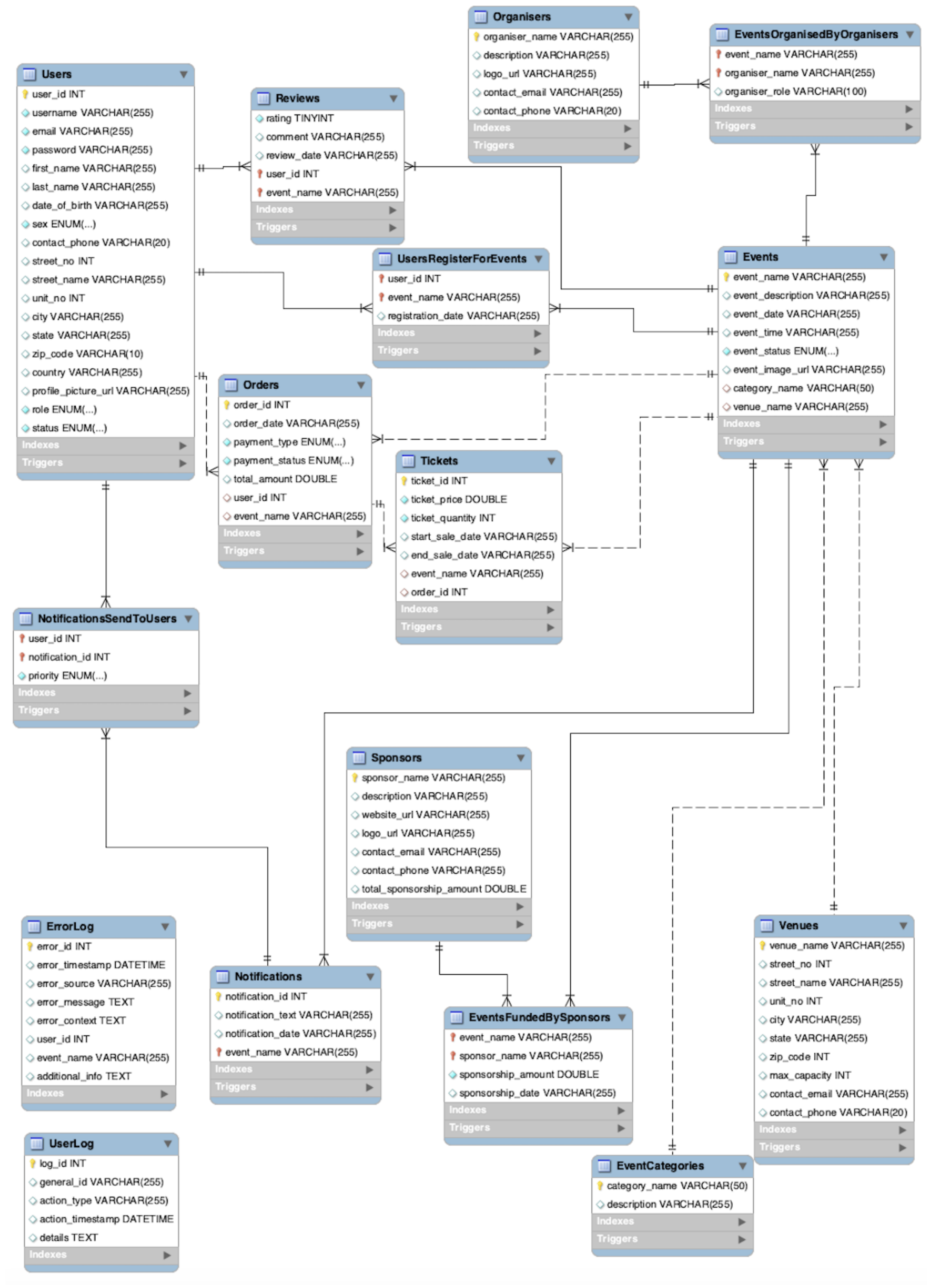
Open another terminal and navigate to folder eventbuzz_frontend
Run on the terminal: “npm start”

NOTE - If after npm start there are any failures due to import missing, delete the node_modules folder and the package_lock file and run “npm install” again.

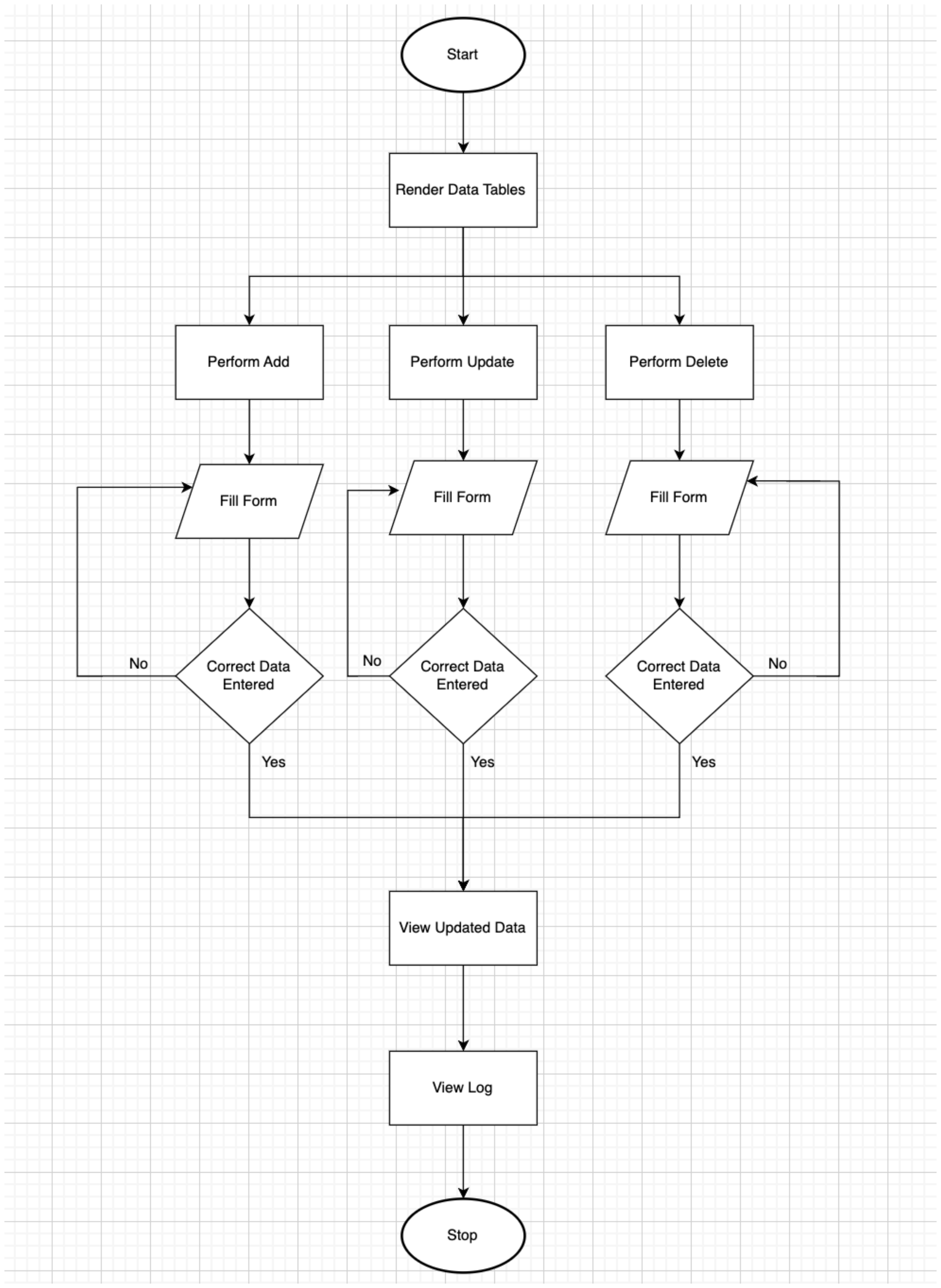
Conceptual Design - UML



Logical Design

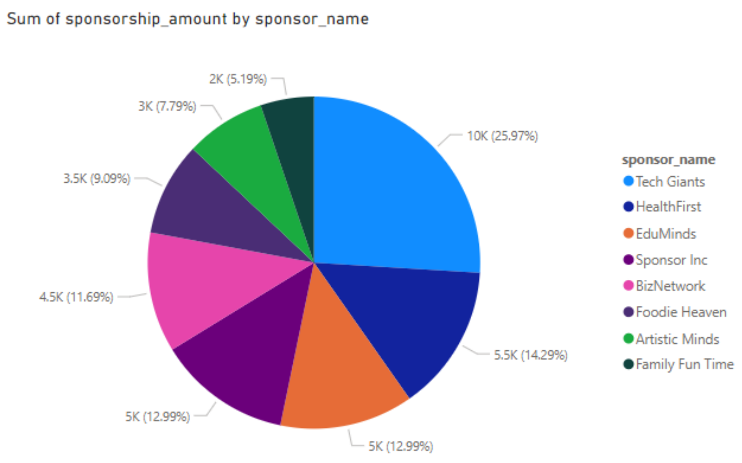


User Flow of the System

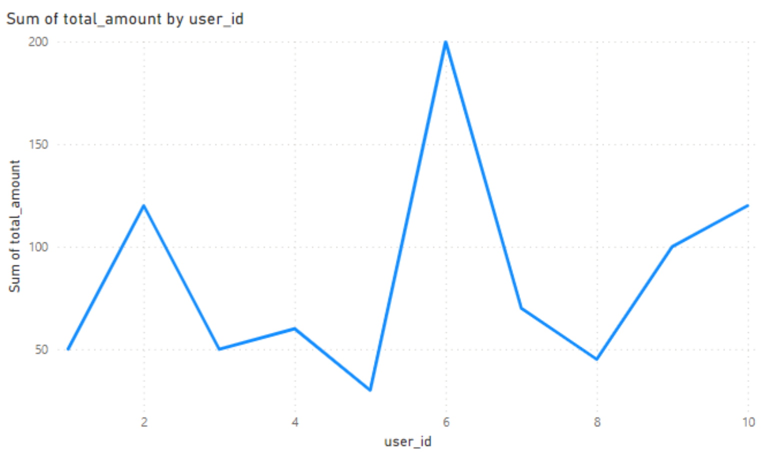


Visual Insights from Power BI

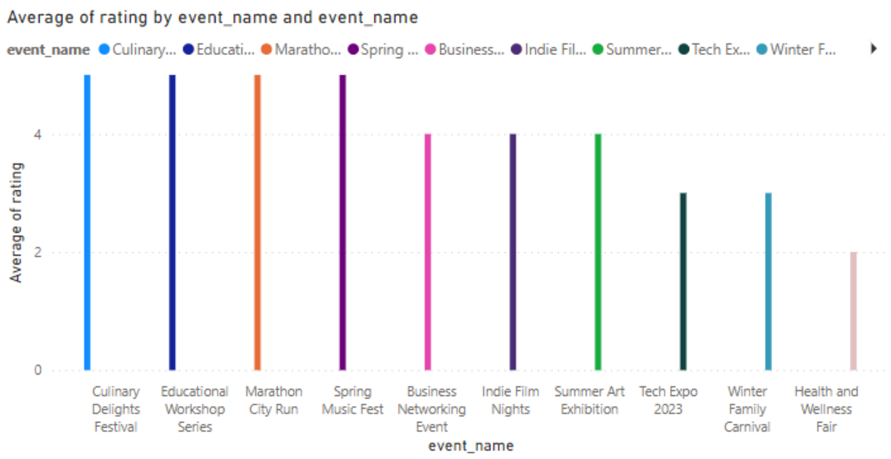
1. Total Money spend by sponsors on the events



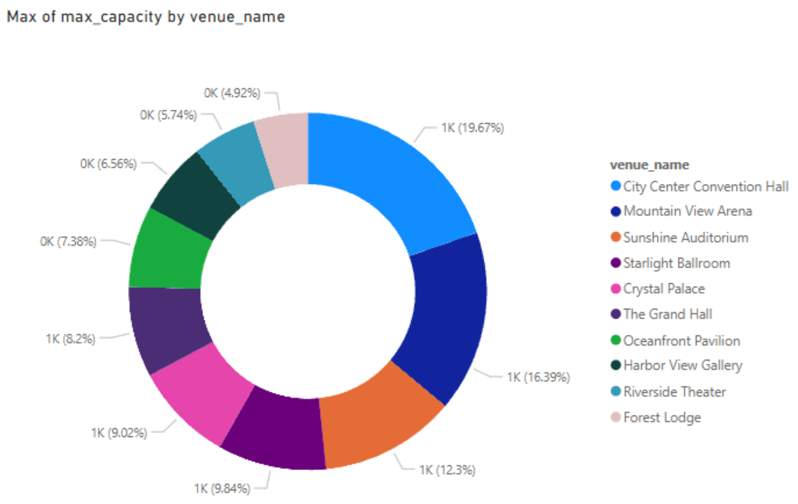
2. Total amount spent by user



3. Average rating of events left by users



4. Max capacity of each venue



Lessons Learned

Technical Expertise Gained

Full-Stack Development: Our experience with EventBuzz enriched our expertise in integrating front-end. back-end and database technologies. Using React.js for dynamic client-side rendering and Node.js with Express.js for server-side logic, we created a seamless user experience.

Relational Database Design: The implementation of a MySQL database not only deepened our understanding of relational database management systems (RDBMS) but also enhanced our skills in schema design, normalization, and SQL scripting. We gained proficiency in handling derived attributes, developing complex stored procedures, and implementing database triggers, which were crucial for dynamic data presentation and maintaining data integrity.

API Development: Designing RESTful APIs was a key learning area. We focused on creating intuitive and efficient endpoints that adhere to REST principles, ensuring our backend was scalable and maintainable.

Version Control: Using Git for version control proved essential for effective collaboration. It enabled us to manage our codebase effectively, allowing multiple team members to work concurrently without conflicts.

Insights

Time Management: The project taught us the importance of effective time management in software development, including setting realistic deadlines and adapting to unforeseen challenges.

Collaboration and Communication: We learned the value of clear communication and collaboration within a team. Regular meetings, code reviews, and shared documentation were key to our project's success.

Data Domain Knowledge: Handling complex data structures and relationships in EventBuzz provided us with valuable insights into data management and optimization techniques.

Alternative Designs and Approaches

Backend Technology Alternatives: While initially considering Python with Flask, we ultimately chose Node.js with React.js for their seamless integration and alignment with our project goals.

Database Options: We evaluated the use of NoSQL databases like MongoDB but found that the structured query capabilities and relational integrity of MySQL better suited our needs.

Frontend Framework Choices: React.js was selected after considering other UI frameworks like Angular and Vue.js, primarily due to its component-based architecture that fits our project requirements best.

Future Work

Expansion of User Interface

Interactive UI Development: Our immediate focus is to evolve the web application from a simple UI performing basic CRUD operations to a more interactive and engaging interface. This development will involve integrating dynamic elements, interactive charts, real-time updates, and a more intuitive navigation structure to enhance the overall user experience.

User Authentication and Role-Based Views: We plan to implement a robust user authentication system, allowing for distinct user profiles such as organizers, sponsors, and general users. Each profile will have tailored views and functionalities, ensuring a personalized experience that aligns with the user's role and interests within the platform.

Planned Uses of the Database

Advanced Analytics and Reporting: We intend to use the database for generating comprehensive analytics and reports, aiding in user behavior analysis, event success metrics, and financial assessments.

Personalized Recommendations: Leveraging user data and interactions, the database will support the development of personalized event recommendations, aiming to boost engagement and user satisfaction.

Scalability for Growing Data Volumes: As EventBuzz expands, we will enhance the database's scalability to manage increasing data volumes, maintaining performance and reliability.

Integration with External Services: Future plans include integrating the database with third-party services for expanded functionalities like payment processing and social media interactions.

Potential Areas for Added Functionality

AI-Driven Insights: Implementing AI to provide more precise event suggestions based on user behavior and preferences.

Enhanced User Interaction Features: Introducing chatbots for customer support, community forums, and interactive features like live polls during events.

Mobile Application Development: Creating a mobile app version of EventBuzz to offer users access to platform features on their mobile devices.

Globalization and Localization: Adapting the platform for different global regions, including language support and region-specific features.

Advanced Security Measures: Strengthening data security and privacy through improved authentication methods and data encryption.

Sustainability Tracking: Adding features to monitor and encourage eco-friendly practices among event organizers and attendees.