Project Writeup

Date: 2025-12-07

Course: CSc337

Group: Elliott Cepin, Iyan Syeed-Miller, Jackson Randolph, and Anthony Joseph

Project Writeup

In this project, we built a three part system for selling venues, hosting events, and buying tickets to events. It is valuable for hosts, because they can reserve venues and host events through the same service; it is valuable for venue owners, because it is convenient for hosts and they are therefore more likely to see listings; it is not much different for event goers, because they are not drawn to services by their convenience but through the popularity of their hosts. Our product is useful and convenient for each of these user types.

Our project has five modules: users, bookings, events, venues, and admins. The user module consists of login, registration, and user data, including current bookings, events, and venues. Any given booking has an associated event. Events cannot have an amount of bookings that exceeds their capacity. Any given event must have a venue. Event capacity is based on venue capacity. There is no limit to how many events can occur at a venue, but only one event can happen at a time in a venue. While admins may login to the site in a similar way to users—they go to a login page and enter their credentials—they do not function like regular users. They cannot own bookings, reserve venues/host events, or sell venues. Instead, they have the ability to update the accounts of other users. For example, if a user needs to refund their booking, a venue needs to be taken down (along with all of the events reserved for that venue), a host needs to cancel their event, and so on, this is the responsibility of an admin user. Because these are critical business operations that will impact many users, they cannot be performed automatically—an administrator is necessary.

The home page of the site contains a basic description of its functionality, and links to the other pages, including login, registration, user profile, and search. Once a user is logged in, they can

access their profile and the search page. On their profile, they will find any information relevant to their account. This includes their username, email, bookings, events, and venue. Along with their email, bookings, and venues, they can see any relevant details. On the search page, users are shown all listings (a listing being either an event or a venue); they can filter listings by type (event or venue). They can also search by listing name, username (of the lister), or keyword. If they search by username or listing name, they must enter an exact match. When filtering by keyword, the given string only must match a substring from a listing or username. The admin user, as mentioned above, can make any change to a users account. This includes updating their username, password, email, and the details of their listings and bookings. Administrators are also able to remove users entirely, but this must be done with caution, as it will impact that users listings.

In order to implement the abovementioned features, some considerations must be made. Because, in production, this site would be handling transactions, account security is important. Losing control of an event via loss of account could be devastating. Thus, necessary security measures have been taken. In production, credentials will be sent to the back-end server via HTTPS. On the back end, passwords are encrypted with scrypt, which is specifically designed to be slow. In production, the hardness can be modulated in order to better defend against brute force attacks. This is much more secure than something like SHA256. Once a user is logged in, they need to be able to stay logged in while moving between sites. To do this, we generate a token for the user that expires after a week. This token is HTTP only, meaning that it cannot be accessed by client-side JavaScript. This is a necessary security measure, because it prevents bad actors from stealing the token and acting as the user. The unique token is used to identify the user between pages without requiring them to log in on each page.

All persistent data is stored in a relational database in order to reduce redundancy. This is best demonstrated by the relationship between events and bookings: the event for which a booking is created is a foreign key in that booking’s relation. There is also a primary-key to foreign-key relationship between events and their venues. Each module has an associated relation. Some

redundancy remains: event’s hold the number of attendees even though that data could be retrieved via query; all bookings point to their event, so an aggregate query could be used to see how many attendees will be at an event. However, it is faster for the system and easier for the programmer to store this data in the event despite its redundancy. Express.js is used for routing due to its simplicity and convenience. The middle wear used for cookies is a package called cookie-parser.