Design Patterns Justification

**Tokens Rather Than Sessions**

In this course, we learned to use sessions with a database to track logged-in users and manage access to features. However, for this project, we opted to use JSON Web Tokens (JWTs) stored on the client side. This simplifies server-client interactions by allowing the client to send a token with each request, acting as proof of authentication and granting access to requested actions. This approach eliminates the need for a session table in the database schema. JWTs also improve scalability since the token itself contains the user’s permissions, enabling different services to validate access without querying a central server. Additionally, JWTs enhance security by keeping personal user data off the server. Our implementation uses a RESTful API, which relies on tokens to authorize requests and provide resources to the client. While we could have used sessions for this project, we felt it was good to expand our knowledge and learn about a technique that is used a great deal in industry.

**Direct SQL Queries Over Stored Procedures**

We chose to use direct SQL queries to the database rather than stored procedures. The reasons for this choice include flexibility in making changes to queries, good functionality with the node.js library, and readability and visibility of code to all team members. Creating and updating specific stored procedures can be more time consuming than writing direct queries. When changes are made to other areas of the code, these queries can be rapidly adapted to new contexts and are easy to test in place and debug. Stored procedures on the other hand, take more time to update and must be dropped and recreated in the database. In addition, changes to a stored procedure could have unintended consequences in other areas of the application where they are used. Because of the node.js database specific library options, a direct query can have all the same capabilities as a stored procedure. These libraries provide many flexible options for working with all types of databases. Another reason we chose direct queries is the ability to have all the code in one place for all developers to see and update as needed. With stored procedures, you often have to refer back to the database in order to understand what they do and how they fit in with the rest of the code, but with direct queries everyone can see exactly what is being called and how it is returned.

**Use of Dashboard Container**

To enhance the website's maintainability and streamline development, we adopted a UI design pattern featuring a static header and footer combined with a dynamic dashboard container. This design choice ensures consistency across the website while providing flexibility for individual pages. The static header and footer serve as the primary navigation and branding elements, appearing consistently on every page.

The dashboard container, positioned between the header and footer, acts as a flexible area where specific page content can be dynamically loaded and displayed. This separation of concerns allows developers to focus on building and maintaining the dashboard content without worrying about duplicating navigation or layout elements. This design also ensures that dashboards built by different developers have the same basic layout and look.

**Pool Rather Than Direct Database Connection**

We choose to use a connection pool instead of direct database connections to reduce overhead for creating connections, manage resources more efficiently, provide for better scalability, and take advantage of automatic error handling. With a direct connection, each individual connection must be initiated, authenticated and closed after use. On the other hand, a connection pool reuses existing connections, resulting in faster connections. The pool also limits the number of connections that can be made to the database at any given time, preventing overload of the database server. When site traffic increases, a connection pool can easily scale to meet demand. The pool can also automatically detect and handle connections that have gone stale, so there is no chance of the application using an invalid connection.