Christopher Clark

SNHU CS-230

29Jan2025

Module 4-4 journal

The client-server pattern is a fundamental tool in software development and sees client requests services provided by servers. The pattern is also very effective when it comes to web-based games that need to run on multiple operating platforms. The interaction between client and server is standardized by using REST-style API which allows for flexibility and scalability. To maintain and update the application efficiently, the REST API acts as an intermediary which ensures that the server and client can evolve independently without affecting one another.

For the server side, the application is developed to take on requests from the clients side. The REST API standardizes communication by defining rules and endpoints that the client then uses to interact with the server. Retrieving game data, updating user scores, and managing user sessions are examples of requests that can be made. The server will process the request and send the appropriate response back, which ensures the client has the necessary data. The stateless nature of the REST API ensures that all requests from the client contains all information needed to process it, which makes the system more robust and scalable.

On the client side, in order to work across multiple environments the application must be developed. Some examples are web browsers, mobiles devices, and game consoles like playstation and xbox. The developers need to ensure the client application can seamlessly communicate with the server through the REST API. This involves things like: cross platform compatibility, user interface design, and API integration. Cross platform compatibility ensures the ability to play on multiple platforms that use different operating systems. Creating a responsive and intuitive user interface design provides a consistent experience across all platforms. Implementing the necessary API calls to communicate with the server and handle data efficiently.

There are a few features that I would add to a game application such as the ability to add friends in game, in game communication and chat system, and a leaderboard page that displays local and global scores. These features would be geared toward the client for ease and accessibility and provides them with information that want. Additionally the security features I would add are the multi-factor authenticator and an encryption on the server. These additions would be beneficial for the server side. This would mostly be important for large game platforms like if the Game Room project were to be played on xbox or playstation. Furthermore, to add more users to the database, user registration and authentication would have to be implemented. This would involve creating user sign-up, login, and profile management. The server would also need to handle storing user data securely in the database. Thorough testing of the application on the various platforms ensures performance and usability. As well as, optimizing the code to handle the specific hardware and software constraints of gaming consoles.

Addressing these aspects will ensure that the web-based game application that I create is robust, scalable, and capable of running on multiple platforms and provide a seamless experience for users.