Module 4-4 Journal: Software Application Requirements

Tyler Ellis

Tyler.Ellis1@snhu.edu

CS-230-H2931 Operating Platforms

Professor Bryant Moscon

November 19, 2023

The client-server architectural pattern is an essential concept in modern software design, especially for web-based applications like the game we are developing in these lessons. The advantages of this framework stem from its ability to break down problems, allowing the client and server sides to function and develop independently while linking through an intuitive interface.   
  
 The client-server design is essential to building web applications that are scalable, maintainable, and efficient. This architecture allows the server to handle game logic, database communications, and state management for the developer's application, whilst the client side focuses on providing the Application's UI and user interactions. Separation is extremely beneficial for compatibility across platforms. By consolidating the game logic on the server, developers ensure all clients, whether web browsers, mobile applications, or desktop applications, operate consistently and efficiently. Clients solely need to worry about displaying the data and processing user inputs. Which can be adapted to the advantages and user expectations associated with each platform.  
  
Server Side  
  
 On the server side, this application has been developed for interacting with the client implementing a REST API method. REST refers to an architectural style that retrieves and controls data through HTTP requests. The server offers an array of endpoints (URLs) which clients can use to perform functions like retrieving game data, altering scores, and managing user accounts. These APIs return data in a non-platform dependent format which is typically JSON or XML, which can be quickly processed by any platform that recognizes these languages. If the API protocol is adhered to, this method separates the client and server, enabling developers to update the server-side functionality or add new features without requiring significant modifications on the client side.

Client Side  
  
 Developing the client side for multiple platforms or environments requires making sure that each client is capable of communicating effectively with the server's REST API and provides a user-friendly interface tailored to each platform. The client application might include a user registration function which transmits user data to a server-side endpoint for account verification and or creation so as to add additional clients to a user database. Something like this is seen commonly in games or applications with real-time multiplayer capabilities, in-game chat, or customizable characters as additional features. To implement these capabilities, client-side APIs and server-side logic would be required.

If The Gaming Room required that the application also be hosted on other platforms like Microsoft's Xbox and Sony's PlayStation 5, you would have to develop client applications for these platforms. These applications must comply with the platform's development standards and regulations, including using the resources provided by its SDKs. The challenge would be to make sure that these new clients can also interact with the REST API that connects to the existing server.  
  
And so, when coupled with a REST-style API, the client-server structure gives developers a flexible and efficient foundation for developing and extending web-based applications across various platforms. This method enables the client and server sides to develop independently. This helps in making it simpler to add new features, support new platforms, and maintain the application's functionality and integrity over time.