4-4 Journal: Software Application Requirements

Michael Wood

SNHU

Course Number: CS-230 Operating Platforms 2024

Emilia Butu

12/01/2024

4-4 Journal: Software Application Requirements

A client-server architecture is a popular method for meeting software requirements and quickly solving development challenges. This technique enables web-based gaming applications to run on multiple operating systems. This is accomplished by separating the game logic and client-side user interface from the game data on the server. This procedure enables the game to be accessed through a variety of browsers, device kinds, and gaming consoles that run on various platforms. Other features that can be offered and implemented using a client-server approach include user chat options and notifications.

While the platform-specific logic is implemented on the client-side, the fundamental application logic and data management can stay on the server-side. The server-side must collaborate with each client to create a multi-platform gaming application. This entails setting up a REST API to manage client requests and facilitate communication between the client and server. Setting up a REST API to handle client queries and enable client-server communication is necessary when creating such a server-side application. The server-side logic includes features like database integration, user administration, and game dynamics, which can be made accessible to the client through RESTful endpoints. The REST API functions as a contract between the client and the server.

It is recommended to adhere to best practices in client-side development, such as using platform-specific frameworks, libraries, or languages to produce an intuitive and responsive user experience. The client-side program must manage user input, render game components, and communicate with the server via RESTful APIs. By expanding the server-side user management capability, more users can be added to the database. Endpoints for managing profiles and registering users may be included in the REST API. An interface for registering new users should be available on the client-side. The input should be verified by the server-side logic, which will then generate a new user record and save it in the database.

An application's client-side development must take into account the various platforms that potential users may use to access it as well as the compatibility of various environments. Screen size, resolution, and input mechanism (such as a keyboard or touchscreen) are a few factors that need to be taken into mind. In order to help developers understand how they want the program to be accessed, including if they will require registration and unique identities and keys for user data to access past states of their game or application, pseudocode is probably going to be most helpful in this section.

Leaderboard functionality, social network connection, and real-time multiplayer capabilities are possible extra features for the game app. To provide the intended functionalities, these enhancements would need additional server-side and client-side development. If a developer wanted to incorporate gaming console platforms into the game room application, they would have to make sure that their platforms and operating systems were compatible.When combined with a REST API, the client-server design offers a strong framework for creating a web-based game application that works on several operating systems. The main game logic and data administration are handled by the server-side, which makes functionality accessible via RESTful endpoints. Platform-specific apps or interfaces that communicate with the server via the REST API are managed on the client-side. The client-server architectural pattern is a flexible and scalable strategy for web-based applications because it enables the effective division of responsibilities and the separate development of server-side and client-side components.