**Client-Server Pattern**

A Client-Server Pattern is an architectural pattern consisting of a client component and server components. The server can be thought of as the hub of a wheel with the clients serving as the spokes. All paths lead to the server. The server processes the inputs from the clients and does work with that information. Processing orders, saving information about users, or returning information requests are a few classic examples of this architecture at work. This architecture is used to solve problems where multiple users, heedless of their OS or geographical location need to access a centralized repository. This pattern fulfills the needs of the client in this instance by providing a hub for users to store game sessions, team compositions and username information in a centralized location. The Server’s database combined with the flexibility of user access levels can provide a suite of interactivity and security tailored to the individual and administrator. Compartmentalization of these roles is of the utmost importance as the application scales.

**REST APIs**

REST stands for Representational State Transfer, which means when a RESTful API is called it transfers a representation of the state of the requested resource. Whether this is calling the Twitter API to get a feed of tweets, or the Instagram API to view a user profile. A REST API consistently provides communication between the server and users accessing that server. The REST API allows for constructing a stable environment for web access coupled with a database for storing user and business information with ease. Through a web browser a user can interact with an API to render information, submit information, and request information. Additionally, the ability to create resources such as users and groups is often present. The REST API Style allows for quickly assembling an interface that triages access based upon login credentials. These ingredients allow for crafting a rich environment that interacts from the server to client in a sophisticated and organized fashion.

**Client Side**

To facilitate access to from various platforms the REST API will need to be tailored to the platform that’s accessing it. Most Operating Systems possess a web browser capability, so this encompasses Windows, Mac, Linux, Android and iOS Operating Systems. If an easier window to access is desired the API would need to be further tailored, an example being gaming consoles or other unique platforms.

Adding users to this database would require expanding upon the access roles. A user could be prompted to store their information on the server with a function that validates their account creation and only allows them a single instance. This method and interaction would be built into the web-application. Presently only Administrators are allowed User creation and deletion capabilities.

Additional features that a client would likely want implemented are various customizations of the user-experience. These could come in the form of badges, achievements, and metrics that a user can see about themselves, their teams, and their interactions with different games on the server. Expanding those metrics to shed insight to a player’s team and game sessions would be a natural progression of this logic. Engagement could also be increased by allowing comparisons of rankings by various demographic or geographical information.

Expanding the hosting of this application to a fourth and fifth client is a logical extension to serving the three initial clients of end-user, server, and database. Examples of a fourth and fifth client possibility could be a gaming console or tablet. If we were discussing expansion to a Microsoft hosted platform such as Xbox or consumer products with access to the play store you could find a way to integrate the web application into the play store, which would be easy to consume for the end-user. If we were discussing hosting on a platform such as Steam, we would start to reach out to their customer service to determine what’s needed to facilitate integrating the web-based game onto their service.