
Draw It or Lose It!

**CS 230 Project Software Design Template**

Version 3.0

**Table of Contents**



## Document Revision History

| Version | Date | Author | Comments |
|---|---|---|---|
| 3.0 | 04/17/2021 | Ryan Sizemore | Fully analysis of desired features and set up has been detailed |

## Executive Summary

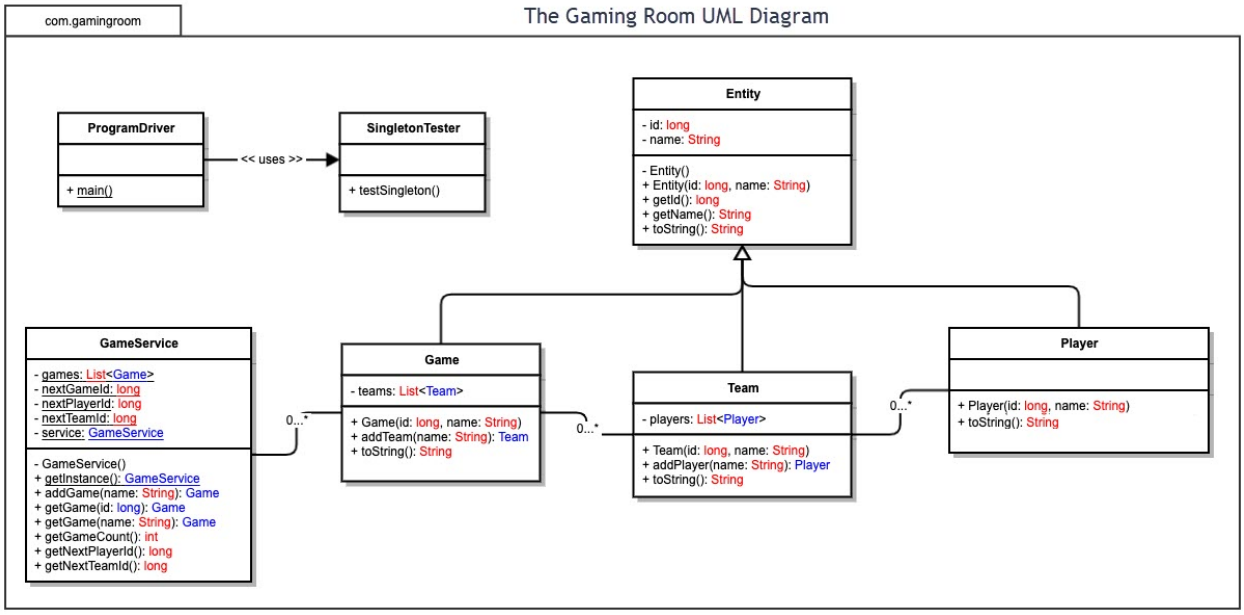
The design used was to ensure there were not multiple instances of the game being played for multiple users. This lays the groundwork for The Gaming Room to build the actual game upon. With the framework laid to ensure the instances of active games are successfully routed, The Gaming Room can build their game atop it.

## Design Constraints

When designing web based apps, design constraints show themselves in browser compatibility, as well as the platform they are being run on. The version ran on an iPhone will be different than that ran on an iPad and different than that ran on Chrome on Windows.

## Domain Model

We have a program driver which drives the actual program, as well as a Singleton class that checks to make sure there is only one instance of the driver being ran. We then have a parent Entity class, with the subclasses Game, Team, and Player. GameService, Game, Team, and Player can all have 0 or more instances to be called upon. There is a list of players, teams, and active games. The player ID is used by team and Entity, the Team is added to the game, and all of that is used by the GameService. All of these converge into the Entity.



**Evaluation**

| Development Requirements | Mac | Linux | Windows | Mobile Devices |
|---|---|---|---|---|
| **Server Side** | Safari is the default browser on macOS and usually runs in to some compatibility issues. It would be wise to explicitly ensure compatibility for Safari. | Linux can run Chrome and Edge and various other browsers. This should make development easy for Linux. | Edge is the default browser for Windows. Windows can run Safari as well as Chrome. Windows. | Chrome and Safari are the two most common browsers on mobile devices. It should be noted that Safari on iOS is different than on iPadOS and macOS, so compatibility should be paid attention to in this realm. |
| **Client Side** | macOS is popular with media developers and students. Mac can run Chrome and Safari. | Linux is a very niche platform with the general public. Therefore it is not as popular as Windows or macOS. | Windows is the primary operating system used across the globe. This will probably be the best place to get return on investment quicker, though being web based, the options are nearly equal. | The world has gone mobile, especially with the worldwide pandemic. I would look heavily into the development for mobile. However, the platform for Apple mobile is different than its desktop class, and specifically for Safari. |
| **Development Tools** | The Apple platform relies heavily on Xcode, the Swift programming language along with Cocoa support. Web development should be universal through JavaScript and HTML. | Linux is a UNIX based system, much like macOS. The Linux kernel is programmed primarily in C and uses a variety of browsers. | Windows uses C as well as Visual Studio to program. Java is a very cross compatible to use between the major operating systems. Windows is the most diverse of the three. | Mobile devices vary between iOS and Android. iOS is very secure and locked down. The only way to develop on platform is to be a registered developer. They use Swift, Android uses Java and Kotlin primarily. IntelliJ is an excellent IDE for Java development. |

**Recommendations**

1. **Operating Platform**: As this is a web based app, I recommend that The Gaming Room do research to find which browsers are the most popular among the age group targeted for the game to be developed. The benefits of creating a web based app, as opposed to a locally ran app is that multiple versions do not need to be created, nor do they have to limit themselves to a single operating system for distribution. By having the application be fully web based, they will be able to make the most use of the system accessing the game as they will be using a native app most likely, meaning an application they know has already been optimized to run.

2. **Operating Systems Architectures**: This is planned to be a web based application. Luckily most browsers play nicely with almost all major operating systems, however The Gaming Room needs to prioritize which browsers they want to support, and if they want to be more mobile or more desktop focused. I encourage The Gaming Room to explore Serverless Architecture options, as those seem to accomplish being cross platform, reducing cost and upkeep, as well as keeping memory management in check. This would allow for expansion in the future in for form of more images to be used.

3. **Storage Management**: Since this is a web based application, the choice of storage for the clients will be cookies to remember their log in information for easy return. On the server side, they will need to store the code that drives the game, this would include libraries and all files required for the program to function, as well as user information such as user name and passwords for validation purposes. We have determined they will need at least 1600MB to store just one copy of the images needed for the game. This does not include any expansions, the raw code, or the libraries and APIs for interface.

4. **Memory Management**: Memory management on a web based app can still be problematic. The game should focus on having few calculations performed on the client side to avoid overwhelming RAM, but at the same time, they cannot overwhelm their servers. This is where the Serverless architecture gets the opportunity to truly shine. For memory management, the logic and computations are offloaded to the accessing machine, while the driving code and libraries are stored on a Function as a Service server. Instances of games would be cleared after the games are ended, reducing the memory usage on both the users and the servers hosting the games. Caching static instances of code on the user device can also result in a more smooth experience for the users.

5. **Distributed Systems and Networks**: The Gaming Room will be relying heavily on server side distribution unless they plan to make this a local app moving forward. Having several servers is going to provide a better user experience and also provide the opportunity to have a fall back if they experience an issue with servers in the future. As traffic picks up, they will want to tabulate how many servers they needs. Using a Mediator approach can assist in ensuring the game proceeds as event driven occurrences, as most games do under the hood. A web based game using HTTP protocols is accessible universally, and should make distributing the game easier than developing dedicated apps for several systems, ensuring they reach as many players as possible.

6. **Security**: Utilize a log in portal complete with password requirements. The information needs to be encrypted with the highest encryption they can afford at first. 128 bit encryption would be strong. Since users will be signing up for accounts, this means that email address as well as password and IP address can be compromised, so ensuring to gather as little information as

needed is a good rule of thumb.  Utilizing compiler side security measures can also assist in the security of the application, and relying on the already present native apps on the operating systems, to take full advantage of the built in security. Systems that run Linux or are various of Linux like Mint are some of the more secure operating systems. The fear when it comes to user privacy brings up the question how The Gaming Room intends to profit off of this game. A web based application may require signing up for a membership, buying a certain number of plays, or a freemium service. If they plan on using advertising for profit, they will also need to consider if they will take the targeted advertisement approach, which will require the acquisition of more sensitive data, which would require more security measures or at the least, disclaimers.