

<Name-of-Software-Application>

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template**](#_heading=h.30j0zll)1

[**Table of Contents**](#_heading=h.1fob9te)2

[**Document Revision History**](#_heading=h.2et92p0)2

[**Executive Summary**](#_heading=h.tyjcwt)3

[**Design Constraints**](#_heading=h.3dy6vkm)3

[**System Architecture View**](#_heading=h.1t3h5sf)3

[**Domain Model**](#_heading=h.4d34og8)3

[**Evaluation**](#_heading=h.2s8eyo1)3

[**Recommendations**](#_heading=h.3rdcrjn)5

## [Document Revision History](#_heading=h.2et92p0)

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| 1.0 | <04/25/21> | Nicholas Boyer | Substantial document |

**Instructions**

Fill in all bracketed information on page one (the cover page), in the Document Revision History table, and below each header. Under each header, remove the bracketed prompt and write your own paragraph response covering the indicated information.

## [Executive Summary](#_heading=h.tyjcwt)

Our client, The Gaming Room, would like to develop a web-based multiplayer game involving players guessing an image as it is being drawn. However, The Game Room needs assistance in setting up an appropriate development environment.

## [Design Constraints](#_heading=h.3dy6vkm)

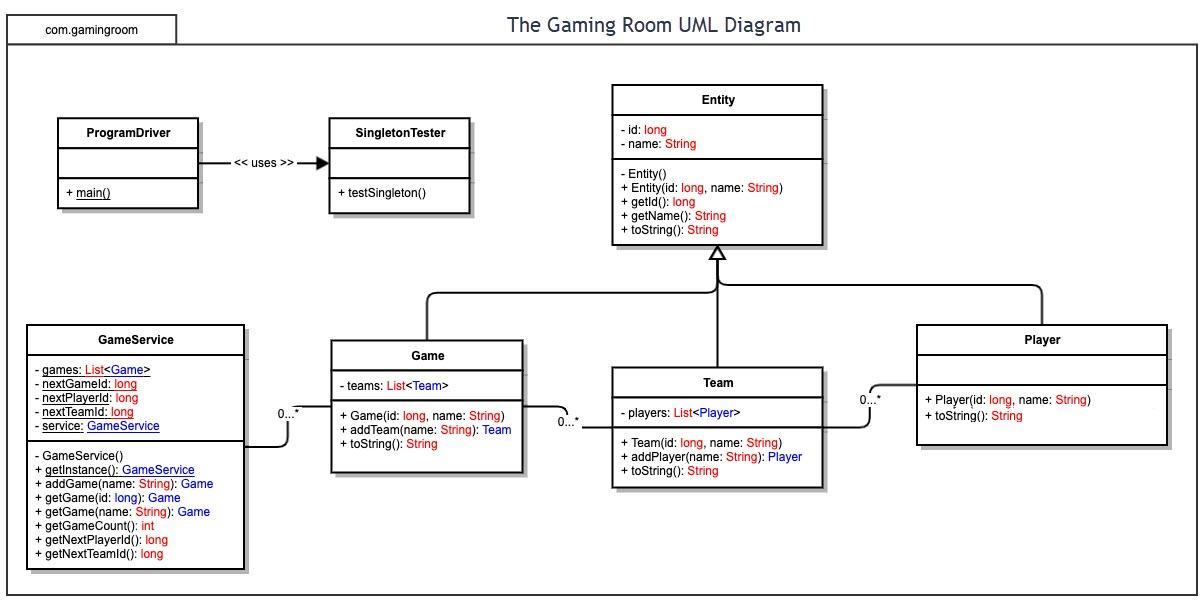
As the game is web-based, users will be sending and receiving data constantly. This means we must take measures to mitigate the effects of latency and poor connections, or else our customers will receive a subpar experience. Additionally, in order to facilitate the game, a client-server architecture will be required. We will have to host and keep our servers online for the game to remain active.

## [System Architecture View](#_heading=h.1t3h5sf)

Please note: There is nothing required here for these projects, but this section serves as a reminder that describing the system and subsystem architecture present in the application, including physical components or tiers, may be required for other projects. A logical topology of the communication and storage aspects is also necessary to understand the overall architecture and should be provided.

## [Domain Model](#_heading=h.4d34og8)

The game is booted by the program driver, which uses a singleton tester. GameService is a singleton class responsible for creating and maintaining instances of the game. The program driver has one static instance of GameServices, meaning it can only use the singleton class when requesting a new game. A game consists of multiple teams, and each team consists of multiple players. Different objects and sets of data in the game are represented as objects.

****

## [Evaluation](#_heading=h.2s8eyo1)

Using your experience to evaluate the characteristics, advantages, and weaknesses of each operating platform (Linux, Mac, and Windows) as well as mobile devices, consider the requirements outlined below and articulate your findings for each. As you complete the table, keep in mind your client’s requirements and look at the situation holistically, as it all has to work together.

In each cell, remove the bracketed prompt and write your own paragraph response covering the indicated information.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| **Server Side** | Difficult and unnecessary due to lack of software. | Excellent for hosting a server. Development is streamlined and there are many tools which support web hosting. Possible tools include Zemap, Webmin, and phpMyAdmin. The larger the our planned server network is, the more efficient it would be to use a cloud hosting service to maintain it. Linux servers can communicate with nearly any other type of server on the internet. | Not as ideal as Linux, but with many tools developed by Microsoft, it would make sense for a team used to Windows OS. | It would not be feasible to host Draw It or Lose It on a mobile device unless we use a P2P system. |
| **Client Side** | Because the game will run on a web browser, development costs will not be extremely high. The Safari browser is well-supported by Apple. Additionally, Chrome is usable on the Mac. | Low cost and maintenance, yet low demand. It is not a priority as only a fraction of players will want to use Linux to play the game. | Low cost and extremely lucrative due to Windows’ large PC market share. It is a necessity to make sure the browser version of Draw It or Lose It is fully functional on Windows. Browser versions for Chrome and Microsoft Edge (and some later versions of Internet Explorer) are necessary to develop. There are no licensing costs associated with developing a browser game. | Mobile devices’ touch screen technology makes them perfect for highly interactive games such as Draw It or Lose It. However, phone applications and HTML browser games are quite different, and a it will take a lot of (well-worth) effort to port the game. We must have developers who are fluent in Javascript, especially the React library. |
| **Development Tools** | The browser version for Mac can be developed alongside Windows and Linux if desired. Testing will have to be done to make sure the game operates correctly on both Chrome and Safari browsers. Support from Apple or testers who have high experience with Apple products would improve development efficiency heavily. | Tools for developing the actual program and hosting the servers on Linux are easy to install and use. Many IDEs are compatible with Linux, such as Eclipse and Atom, are available on Linux. | As a web browser game, its appearance will be dependent on HTML and CSS, while Javascript, which runs on the client browser, will be responsible for the client side’s function. IDEs such as Atom are excellent for developing Javascript. | Most modern iphone apps are written using the Swift language. Android phones largely run on Java. Again, because these executables are functionally different from our planned browser game, it will take effort to rewrite the game in Javascript. |

## Recommendations

Analyze the characteristics of and techniques specific to various systems architectures and make a recommendation to The Gaming Room. Specifically, address the following:

1. **Operating Platform**: I recommend that we pursue a linux-based cloud hosted server. Linux operating systems are excellent for hosting servers due to their low overhead, ease of use, and many great tools.
2. **Operating Systems Architectures**: We will use the Ubuntu distribution of Linux in order to host our servers. We will pay for hosting centers across the nation, especially around high-traffic areas, to reduce any bottlenecks in throughput and to decrease the odds of catastrophic data loss.
3. **Storage Management**: Usernames and passwords will be stored and encrypted on our servers. We will develop online browser-based clients for multiple operating systems, such as Windows, Mac, and mobile devices, so that our game can be played across many platforms. These assets will be sent to our clients and cached on their local machines. As most data associated with the game will not be used at once and we do not wish to send redundant data to the clients, we will make sure each browser’s caching behavior works correctly with their operating system’s file storage system.
4. **Memory Management**: Game assets, when in immediate use, will be handled by the client operating system. Because each operating system has a different way of storing and retrieving files, it is important that we develop each client with its OS in mind. We do not want to interrupt with the operating system’s ability to multitask, nor do we want to cause hazardous malfunctions on the client computers.
5. **Distributed Systems and Networks**: Because Draw It or Lose It will be playable across the world, it is important that we have servers in many areas. This is to reduce bottlenecks when clients are connecting to our servers and to increase ping times. When it comes to video games, ping is incredibly important, as fractions of a second can determine the winner and loser of a game. Additionally, it would be optimal if we were to host on servers connected to an on-site generator, so that local outages do not produce hours upon hours of downtime and lost revenue.
6. **Security**: Security is difficult to implement, but thankfully, there are many libraries and services available. When it comes to security, you do not want to reinvent the wheel, especially when it means risking your customer’s data. To ensure security, we can encrypt communications between our servers and clients, as well as keep user data encrypted. We will use standard protocols, such as https, so that our users are safe no matter what operating system they choose. We will also use authentication/authorization technology to reduce our vulnerability to malicious attacks.