

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 3.0

## Table of Contents

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

[**Table of Contents**](#_heading=h.30j0zll)2

[**Document Revision History**](#_heading=h.3znysh7)2

[**Executive Summary**](#_heading=h.2et92p0)3

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

[**System Architecture View**](#_heading=h.3dy6vkm)3

[**Domain Model**](#_heading=h.1t3h5sf)3

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

[**Recommendations**](#_heading=h.17dp8vu)5

## [Document Revision History](#_heading=h.3znysh7)

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| 1.0  2.0  3.0 | 7/21/2021  8/1/2021  8/16/2021 | Enrique Zarate  Enrique Zarate  Enrique Zarate | Additions to the Executive Summary, Design Constraints, and Domain Model have been added before building of the program begins.  Added evaluations of each different operating system and their pros and cons of each. Attempting to identify which tools and languages may be used for this multi platform application.  Recommendations added based on the needs of |

**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.2et92p0)

The staff at The Gaming Room would like to develop a web-based game that will have one or more teams in each game and each team will have multiple players. Only one instance of each game shall exist in the memory at a time. Unique identifiers need to be given to each game, team, and player to allow for only one instance of each of those.

## [Design Constraints](#_heading=h.tyjcwt)

This web-based program will need to be able to be accessed on multiple devices. Java will allow for use across multiple browsers and devices. With the requirements set by the client, The Gaming Room, a singleton design pattern will be used to create the unique instances of each item and to prevent duplication of any users, games, or teams. For web-based games, the users will need a stable internet connection and depending on their targeted audience’s age, this may need to be re-thought. From the aspect of the gaming side, lots of testing may be required to test the timing and accuracy of the rendering image in order to allow for fairness among all players.

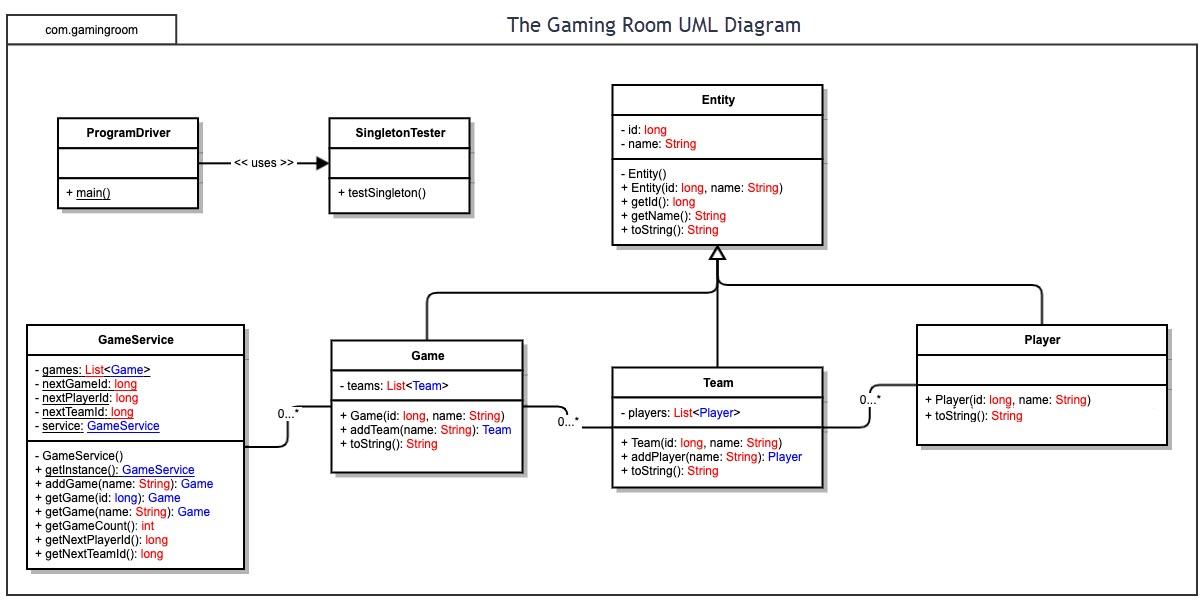
## [System Architecture View](#_heading=h.3dy6vkm)

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.1t3h5sf)

Only one instance of each game may exist in the memory at a time, for this reason, a singleton design pattern is being used and is demonstrated in the UML Diagram below. The GameService() function itself is a private class and can only be called upon by itself. Many instances of a game, team, and players can be created but each will be a single instance of itself with a unique identifier also shown as the “Entity” class. Each entity (game, player, team) will have its own unique ID number and it’s own name which will allow the system to look up usernames or team names that may have already been taken. The GameService class will be responsible for keeping track of current ID’s, Team Names, and Player IDs to ensure only one instance of each is created.

The Game, Team, and Player class are all Inheriting traits from the Entity class so that way we can ensure that each of these items are unique and there cannot be a duplicated instance of any of these items.

****

## [Evaluation](#_heading=h.4d34og8)

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** | When it comes to hosting a web-based software, Mac generally isn’t the first to be picked. Although Apple has some solid products, generally speaking, they are not as accessible to others with their high price tag. Not only that but the ability to customize them generally is low as well. They tend to have high computing power which may make up for it but it is not advised for hosting a web-based game on mac. | Linux tends to be one of the top picks for server hosting capabilities of web based applications. Although there is less of a pool of people that tend to be familiar with this operating system, it still tends to be the cheapest for hosting services. While this is the case, since the game will be cross platform, the clients do not need to match in OS to access the server contents as long as it is set up correctly. | Windows operating system for servers tends to be one of the highest as well. It tends to be less popular because of its cost compared to Linux. Another upside to this is there is a large group of people who are familiar with Windows since it tends to be one of the most familiar amongst people and the most accessible. This would be a second choice. | Very similar to the Mac, hosting tends to not happen on these platforms. Although, if we did want more of the matching between the server and client side, if the transition to move over to mobile only is made, that is when it may be easier to choose a mobile OS as a host for the web-based content. |
| **Client Side** | A Mac specialist team may be needed as these can be very specific with how the client side is built. Not only that but Mac’s tend to have many updates all the time and these updates can interfere with current versions of the game that are built. | Similar to Mac in where you may need an individual or a small team around this, I believe there are way less Linux users than the other types of operating system which would mean that the cost for this should be significantly less than the other developers needed for the other operating system. | Windows will probably be second on the list for priority of a team. Although Windows does have some updates, the updates tend to allow for backward compatibility for quite a bit of time before becoming incompatible with other updates. I would put this on the same level with mac in terms of possible equal number of users but less work intensive than the Mac OS. | According to my experience, the mobile team will be the largest. It is the demographic that our goal tends to focus on as the number of people with cell phones is going ot be the largest. The number of people that play these types of games may also be on mobile since there is a large wave of children that tend to be getting more phones nowadays. With how many different operating systems and compatibilities there are, it is definitely expected that this team would be the largest in order to cut down on the amount of time it takes to finish development. |
| **Development Tools** | With most Mac products, the go to for development tends to be in Swift. Mac also came out with a very specific program that can be obtained through the App store called XCode which allows for the development of products specific to Mac products and Mac’s mobile products. | I have the least amount of experience with Linux, but I believe that C++ or Java would be the way to go here. Especially since we are most likely going to use the Linux OS for hosting a server for the game. | For Windows, C++ will allow for most of the graphical kind of that the users need to experience and that us allow us for the memory storage and moderate security needed to prevent hacking. IDE like visual studio will help with this. | A mixture of languages will be needed for this depending on the mobile service. As talked about earlier, if they are Apple Products, specific languages like Swift may be used. Otherwise, a general approach can be taken with a survey of all the different operating system to be used and finding one of the most compatible languages between each of the operating systems. Polyglot programming will really be useful here. |

## 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**: <Recommend an appropriate operating platform that will allow The Gaming Room to expand Draw It or Lose It to other computing environments.>
2. **Operating Systems Architectures**: <Describe the details of the chosen operating platform architectures.>
3. **Storage Management**: <Identify an appropriate storage management system to be used with the recommended operating platform.>
4. **Memory Management**: <Explain how the recommended operating platform uses memory management techniques for the Draw It or Lose It software.>
5. **Distributed Systems and Networks**: <Knowing that the client would like Draw It or Lose It to communicate between various platforms, explain how this may be accomplished with distributed software and the network that connects the devices. Consider the dependencies between the components within the distributed systems and networks (connectivity, outages, and so on).>
6. **Security**: <Security is a must-have for the client. Explain how to protect user information on and between various platforms. Consider the user protection and security capabilities of the recommended operating platform.>