

DrawOrLose

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template**](#_l6ti7uoag22u)1

[**Table of Contents**](#_30j0zll)2

[**Document Revision History**](#_grjogdjh5fi8)2

[**Executive Summary**](#_sbfa50wo7nsh)3

[**Design Constraints**](#_2et92p0)3

[**System Architecture View**](#_ilbxbyevv6b6)3

[**Domain Model**](#_8h2ehzxfam4o)4

[**Evaluation**](#_2o15spng8stw)5

[**Recommendations**](#_m8aleynsvzvc)6

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 05/22/2021 | James Porter | Updated the document to include requirements that the client requested and needed. |
| 1.1 | 06/01/2021 | James Porter | Updated document using feedback from instructor. |
| 1.2 | 06/06/2021 | James Porter | Updated the documents security section to include recommendations for what security features should be used for this software. |
| 1.3 | 06/15/2021 | James Porter | Updated document from recommendations from the instructor from week 5 submission. |
| 1.4 | 06/17-19/2021 | James Porter | Made recommendations and final updates based on findings |

**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](#_sbfa50wo7nsh)

The Gaming Room is looking to develop a web-based version of the game Draw It or Lose It (this is loosely comparable to the TV game show *Win, Lose, or Draw*) that is currently only available as an Android app. This document serves to show an outline and plan to move forward with the development of this game. Additionally, this document shows the UML diagram and evaluations of the pros and cons of using different operating systems for development. At the end of the day, it is recommended to develop this program with Linux as it allows for the use of universal languages and allows for quick development.

## [Design Constraints](#_2et92p0)

A game must be able to have one or more teams involved at a time, and each team must be able to have multiple players assigned at a time. Each game and team names must be unique to allow them to be easily searched and checked by the user to see if a name is being used. Lastly, only one instance of the game can exist at a time; this can be accomplished by creating unique identifiers.

## [System Architecture View](#_ilbxbyevv6b6)

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](#_8h2ehzxfam4o)

The ProgramDriver contains the main class and will run the program. It will use the SingletonTester class to ensure that only one instance of the game room exists. The Entity class will have two variables id and name which will be inherited by the three child classes. The entity class is built using the singleton pattern which only allows one instance of the Entity class to be created. The Game, Team, and Player classes all inherit the id and name variables from the Entity class. The Game class has a list of the teams in a game using the game id. The Team class has a list of the players on each team using the team id. The Player class just contains information on the individual players on each team. The GameService class is created using the singleton pattern as well and this means that only one instance of the GameService class can be created. This class contains the list of Games as well as the nextGameId, nextPlayerId, and nextTeamId. Each GameService class can have zero to as many as desired instances of the Game class; each Game class can have zero to as many as desired instances of the Team class; and each Team class can have zero to as many as desired instances of the player class. This is to say that the game service can have multiple games in the list, the game can have as many teams as desired in the list, and each team can have as many players as desired on each team.

****

## [Evaluation](#_2o15spng8stw)

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** | * More expensive (To purchase) * Easy to Learn * Reliable | * Lowest cost to own * Very time expensive * Easy to learn at first * Difficult for anything more complex * Easy to maintain | * Prone to malware * Inexpensive * More upkeep needed | * Extra software necessary to run and use * Low cost * Very limited functionality * Very specific individual needed to run |
| **Client Side** | * Fewer web browsers available than other OS * Has browsers that would work across multiple platforms * Safari is the default web browser * Relatively easy to download and install * Most browsers free of charge | * Wide variety of browsers available * Installation and download more intensive than other OS * More customizable browser experience | * Wide variety of browsers available for use * Microsoft edge (or internet explorer on older versions) is the default * Other browsers available for download * Most browsers free of charge to download | * Very limited mobile browsers * Limited depending on phone type * Lighter browser versions available |
| **Development Tools** | * Multiple IDEs Available for download * Extra steps are necessary to download/install necessary IDE and supporting Toolkits * Support available for Java IDEs using MacOS | * Multiple IDEs available for download * Extra steps required for to download/install necessary IDE and supporting Toolkits * Some support available for Java IDEs using Linux | * Multiple Java IDEs available for download * High levels of support for Java IDEs using Windows * Relatively simple download/install (Fewer extra steps necessary) | * Development on mobile devices most likely impossible * If it is possible, it is very limited in its capabilities |

**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 using Linux to develop the program as it provides the most benefits for the client. It is low cost to own and easy to maintain. Linux also offers some unique benefits that MacOS and Windows does not offer (discussed below). Additionally, Linux has the capability of running Eclipse and NetBeans (among many others), two powerful Java IDEs used for program development. For Draw it or Lose it I recommend using Eclipse, because it is the program that has been used to develop the software so far. Eclipse is also free and offers support on multiple different websites.
2. **Operating Systems Architectures**: The Linux operating system (OS) has a unique way of operating and is set up in layers. The inner most level consists of hardware (inner workings) that run the computer. The next layer out is the kernel which is mainly responsible for most major activities for the Linux OS. These kernels interact directly with the hardware (The inner most layer) to make the computer run properly. System libraries and special functions, often called the shell, are the next outermost layer (after hardware and the kernel) also help these kernels run and allows for specialized tasks to run. The last layer is the interface that allows the user to interact with the kernel to run the hardware. The shell is an important part of this structure as it adds a layer of protection from the user when it comes to the inner workings of the computer.
3. **Storage Management**: Linux OS represents almost everything with a file and these files are saved on a storage device (typically SD or HD). Partitions are used on these drives as well, but this causes some issues as these partitions are assigned on bootup. A work around to this is to assign a more persistent/permanent partition on the device. These drives would be used to store the extra images that aren’t being used in the current or next game(s). This also allows the user to import more photos as necessary to increase the variety of games allowed. This storage would be slower but is cheaper to buy so you can get more storage at a smaller price.
4. **Memory Management**: Linux OS uses demand paging to process virtual memory. This means that whatever command is executed, the file containing it is opened and mapped to process the memory. If it needs to access another memory location within this memory location it accesses that location and maps it like it did the first. In the situation comes up that there is no physical space, files are swapped out. This is where the files that are going to be used for the current or next games to be played. This memory is a lot quicker to access, but is more volatile and upon losing power, this memory is lost. Additionally, buying memory costs a lot more than storage when it comes to price per GB.
5. **Distributed Systems and Networks**: Linux allows the most accessibility between networks due to its high level of customization. This customization allows the developer to the needs of the client, in this case so multiple computers can “talk” to each other over a network. Linux can also be used in a Server-Client relationship which can be used to give access to the game to individuals across platforms. This is essential to running the online game Draw it or Lose it as the game must be able to be played across multiple computers/networks.
6. **Security**: As stated above, using Linux to develop this software is beneficial to the security, because of the layers of the operating system. Additional security measures can be used to increase how secure the software is. One feature that is a must for this software is encryption. Important data, such as usernames and passwords, should be encrypted for increased security. It is important to limit who has access to the encryption and decryption keys (this topic is referred to as authorization and is discussed later). Another tool that should be used is called authorization and authentication. These two processes work together and state what each user can do (authorization) and require the user to input username and password to ensure they are who they are (authentication). Lastly random audits could be completed to ensure that only users that allowed to make changes are making changes.