

<Name-of-Software-Application>

# **CS 230 Project Software Design Template**

Version 1.0

## 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.2s8eyo1)3

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

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

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 11/12/2022 | Mikayla-Joy Botha | Modifications and Additions |

**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)

I am a part of the Creative Technology Solutions (CTS) as a tech consultant. Right now one of our clients, The Gaming Room, would like to upgrade and expand their game called Draw it or lose it. This game was popular in the 80’s, and based on a tv game show named Win, Lose, or Draw. The Gaming Room would like to expand their game by making it versatile to multiple platforms. Hence in this document, we will evaluate which is the best possible way to go about this.

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

## Needs to be web-based

## Serves multiple different platforms, not just android

## App must have the ability for multiplayer mode with teams consisting of more than one person assigned to them.

## Both game and team names should be creative with requirements to verify if the name already exists when submitting a team name.

## Only a single game instance can be exist in the memory whenever, hence this can be achieved by incorporating unique identifier for every game, team or player instance

## [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)

Primarily, The Program Driver Class will create a single instance of GameService with some game data from Game #1 and Game #2. After that, The Program Driver class will create an object called tester for another class called SingletonTester that is responsible for proving that there will only be one instance. Furthermore, there is an Entity class which is the base parent class containing attributes and behaviors which will be inherited by the child classes. These child classes will be Player, Team, and Game.

The Entity parent class contains attributes such as id, and name that will be passed on to extended child classes. Then for the connected GameService class, it will perform a singleton service in favor of the game engine, construct a new game instance, return existing id’s or names, return the game instance with a specified id, and will return a game with a specified name.

**"The Gaming Room UML diagram. The top of the diagram is labeled as com dot gamingroom. Test boxes are placed in two layers. The first layer has three text boxes and the second layer has four of them. In the first layer, the 'ProgramDriver' textbox points to 'SingletonTester' textbox. The 'ProgramDriver' textbox contains the text 'asterisk main round brackets.' The 'SingletonTester' textbox contains the text 'asterisk testSingleton round brackets.' The arrow between these two text boxes are labeled 'open two angle brackets uses close two angle brackets'. In the second layer, there are 'GameService', 'Game', 'Team', and 'Player' text boxes. The 'GameService' textbox has texts arranged in two layers. The first layer contains games colon List open angle bracket Game close angle bracket, nextGamesId colon long, nextPlayer Id colon long, nextTeamId colon long, and service colon GameService. The second layer contains GameService round brackets, getinstance round brackets colon GameService, addGame open parenthesis name colon String close parenthesis colon Game, getGame open parenthesis id colon long close open parenthesis colon Game, getGame open open parenthesis name colon String close open parenthesis colon Game, getGameCount round brackets colon int, getNextPlayerID round brackets colon long, and getNextTeamId round brackets colon long. The 'GameService' box is connected with the 'Game' textbox with a line labeled 'zero dot dt dot asterisk'.  The 'Game' textbox also contains text in two layers. The first layers contains the text teams colon List open angle bracket Team close angle bracket. The second layer has Game open round bracket id colon long comma name colon String close parenthesis, addTeam open parenthesis name colon String close parenthesis Team, toString round brackets colon String. The 'Game' textbox is connected with the 'Team' textbox with a line labeled 'zero dot dt dot asterisk'. The 'Team' textbox also contains text in two layers. The first layers contains the text players colon List open angle bracket Player close angle bracket. The second layer has Team open parenthesis id colon long comma name colon String close parenthesis, addPlayer open parenthesis name colon String close parenthesis colon Player, and toString round brackets colon String. The 'Team' textbox is connected with the 'Player' textbox with a line labeled 'zero dot dt dot asterisk'. It contains the text Player open parenthesis id colon long comma name colon String close parenthesis and toString round brackets colon String. The 'Game', the 'Team, and the 'Player' boxes point to the 'Entity' textbox in first layer. The 'Entity' textbox contains text in two layers. The first layer has the text id colon long and name colon String. The second layer has Entity round brackets, Entity open parenthesis id colon long comma name colon String close parenthesis, getId round brackets colon long, getName round brackets colon String, toString round brackets colon String.**

## [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** | I haven’t had much experience with Mac on the server side. However, I will say one advantage is the Caching Server for content. This speeds up Apple System updates, as well as data consumers will save in iCloud from content on their devices. Another advantage is MacOS quality File Server, it is a quick way to share content between users. | Linux is best equipped for web-based software applications, as it utilizes an open-source OS. This makes it flexible for smaller projects. | Out of all the OS options listed, Windows is by far my least favorite. Every time I have used Windows OS for hosting a web-based software application, it has been exceedingly slow. | Mobile devices advantages or disadvantages depend on whether its OS is Android or Mac. Generally overall though, the clear advantages are portability, and convenience. A disadvantage could be the limited screen view size. |
| **Client Side** | Since MacOS utilizes the client side, it focuses on one user while also having more than enough power for multiprocessing. This raises the quality experience overall, which makes the product more expensive. This OS uses XCode. | For Linux, it utilizes Ubuntu for the operating system. Linux can support a multiple client system, more cheaply than alternative options. | Mainly it seems that HTML pages talk to clients. Which understandably makes the system slower. Thus because performance is lower, it comes at a cheaper price. | As for development considerations, if its an Android system, it will use Android Studio. On the other hand if IOS will utilize XCode. Android runs slower, but is cheaper. The iOS is faster but more expensive. |
| **Development Tools** | In comparison to other operating systems, MacOS only uses swift to program their OS. Swift is used for all of their products including iOS, Apple Watch, Apple TV, and Mac. | The most relevant programming languages that linux uses are Python, C and C++. A few tools I've used are Ubuntu Linux, VSCode, and Draw.io. | Languages most common for programming with windows are Java, HTML, CSS and JavaScript. Windows generally uses Visual Studios to develop and build. | As discussed earlier, iOS will use Swift while Android uses Java for programming languages to develop. |

## 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**: Among reviewing different operating platforms, I strongly suggest choosing to work with Mac iOS for Draw It OR Lose It.
2. **Operating Systems Architectures**: Mac OS may be a little more expensive, but is fast and reliable. Mac OS is harder to hack, and thus does not frequent viruses as much as Windows OS.
3. **Storage Management**: An appropriate store management system to be used with Mac OS is iCloud. With iCloud, it is possible to securely store your data that is accessible on any Apple device (and even non-Apple devices) with an Apple sign in.
4. **Memory Management**: Mac OS uses memory management techniques for Draw It or Lose It software, cheaply containing all photos, files, notes, all data in one Cloud across any apple devices with the same login. The iCloud memory bank, also can easily be backed up so if you lose any files they can be retrieved.
5. **Distributed Systems and Networks**: Knowing that the Client would want Draw It or Lose It to communicate between various platforms, this also favors Mac OS. That is because, as listed to Memory Management for any apple device they can all easily be connected to communicate if each one has been logged into with the same apple ID login.
6. **Security**: Essentially, with using iCloud the user information is protected because it is stored in an encrypted way, which makes it more secure. Any apple device incorporates forms of encryption through different safeguards for user protection. You can even wipe your apple device completely from a different remote apple device, if they are connected if yours is lost.