

Draw It or Lose It!

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template** 1](#_Toc115077317)

[**Table of Contents 2**](#_Toc115077318)

[**Document Revision History 2**](#_Toc115077319)

[**Executive Summary 3**](#_Toc115077320)

[**Requirements 3**](#_Toc115077321)

[**Design Constraints 3**](#_Toc115077322)

[**System Architecture View 3**](#_Toc115077323)

[**Domain Model 3**](#_Toc115077324)

[**Evaluation 4**](#_Toc115077325)

[**Recommendations 5**](#_Toc115077326)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 03/18/2023 | Matthew Alexander | Initial revision. |
| 1.01 | 04/15/2023 | Matthew Alexander | Slight edits to the OS table. |

**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 client requests that each instance of their game is unique and can only be ran once. They wish for this game to work on multiple client types, including mobile, laptop, and desktop computers.

## Requirements

*<* Please note: While this section is not being assessed, it will support your outline of the design constraints below. *In your summary, identify each of the client’s business and technical requirements in a clear and concise manner.>*

## [Design Constraints](#_2et92p0)

The main design constraint of a web-based game is that you are limited to web-browser functionality when accessing hardware. This can be made easier by handing off the duty to an app-based client. Another design constraint is the server. The server must be able to handle instances of the game running to every player.

## [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 UML diagram below shows that the entity is a supertype of all the entities below it, like the player. Each subtype will inherit attributes from the entity class and then the GameService is responsible for managing the entities.

**"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](#_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** | Macs can make decent servers on a small scale, but have issues with limited hardware. They lack expandability and will not serve well on the larger scale. | Linux, I think is the best for this. It is completely configurable and customizable to the needs. The only down-side is that it isn’t the most user-friendly and configuring might be trickier. | While Windows is not the best, it can do the job pretty well for most use cases. It is much more user friendly and offers simpler setup, but might provide less options for configuration. | Mobile devices are not suitable to be a server. |
| **Client Side** | MacOS is an interesting client to make because it is the most Unique OS. It has a lot of requirements for an app to run. Also with the recent switch to ARM based chips, you have to take that into consideration. | Linux as a client side is good because the people who are using it, usually have a good idea of what they are doing. They will be able to get their client to work. But the low user count compared to mac and Windows might make it seem not worth the hassle. | Windows is by far the easiest thing to develop for. Has low cost, the expertise doesn’t need to be that high, and you can deploy straight to the machine the developing is happening on. | Mobile clients are interesting because they have special rules that Apple and Google have set up for allowing apps on their app stores. Also with mobile, you are limited to touch based input, and you need to innovate to make something designed for keyboard and mouse work on mobile. |
| **Development Tools** | MacOS and windows are the two most common operating systems and it would make sense that Mac would be one of the OS’s used for development since it would be easier to deploy for it. Also needed for iOS app development. | Linux probably would not make the best development environment because you wouldn’t be able to deploy and test on Mac or Windows inside of it. Could be ok for Android app development. | Windows is probably one of the better choices because of its versatility. It can develop for itself, Android, and Linux using the WSL feature. It is also the most familiar OS to most people. | Development tools just shouldn’t be used on mobile. Maybe good for QA and catching errors on real hardware, but that is about it. |

## 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 think the best operating platform for this app would be the Linux operating system because it has the most configuration and is the best for backend work.
2. **Operating Systems Architectures**: The architecture would be x64 as it is the most common architecture and the easiest to use.
3. **Storage Management**: Storage should be managed with time. For example, users usually like to save replays of their games. At the end of the game, all data to the game should be saved for players to access for a certain amount of time. After that time, the files are deleted.
4. **Memory Management**: Using the singleton patterns, we will prevent duplicate instances of the game, which will help cut down on memory usage. Linux as a system also uses the least memory out of any OS currently available and is very lightweight.
5. **Distributed Systems and Networks**: The software clients must be able to read and send data to the server as consistently as possible. Since all platforms are able to connect to the internet, they will all be able to communicate with each other though the server.
6. **Security**: Security must be implemented in any app made. Like keeping the app surface level and not requiring administrator privileges will prevent a hijacker from getting users information though the game. The server must also check all data coming in to make sure it is coming from the players and not a malicious attack pretending to be a player.