

<The Gaming Room>

# **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 | <11/12/2023> | <Caleb McManus > | <The completing of this document> |
| 1.0 | <11/26/2023 | <Caleb McManus> | <The Evaluation> |

**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 is known as The Gaming Room, where they desired a Draw It or Loss It game environment which is now what I’ve provided in coding.>

## Requirements

*<* I was required to code in the environment while making it very easy to interface with as the client has little experience with the environment and coding in general*.>*

## [Design Constraints](#_2et92p0)

<Needs to have the ability to be malleable; by which I mean it needs to be able to add teams and players as needed as the game is progressed. Along with it, the system needs to be easy to understand for further development.>

## [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 Program driver and the tester are separated from the rest of the classes as they are there on bubble that interacts with each other and then incorporates the other bubble of classes. As for the rest, GameService and Game are interacting with each other to build out the system of the game. Game also shares with Team and Player under the extension of Entity as the parent class that passes down the shared functions.

**"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** | Mac has a big strength that also happens to be a weakness of it. Mac has a great developer friendly environment that allows rapid changes and deployment because of the robust tools. However, its it’s biggest weakness as it is pretty much only runnable on Apple hardware, which tends to be more expensive than other platforms. | Due to the customizability of Linux, Linux servers tend to be extremely well centered and built for the task in mind. This allows developers to run the servers in more stable variants with little issue of having conflicts. Overall, it can be as lightweight or heavy weight as you need it to be. Unfortunately, it has the issue of being unstandardized that it is difficult for developers to just jump in. | Windows’ biggest advantage is that it is widely supported and built for. Users and developers tend to use windows machine, and by proxy allows them to feel more comfortable developing and running it. Unfortunately, the issue of running into conflicts with the OS does tend to slow development and bring up server down time. | Mobile is highly unrecommended as it is extremely difficult to develop for mobile apps as it would be needing to be built completely from scratch, which is fair less viable than all other options mentioned.  What it can do, the others can tend to do better.v |
| **Client Side** | As mentioned above, due to Mac being exclusively run on Apple hardware, it tends to be far more expensive than alternatives. As for ease of use and learn time, it would be short to moderate depending on the user’s knowledge of windows which would just be slight alterations. | Linux is extremely desirable in terms of cost as due to its open-source nature; it is free to start and is openly developed across the world. Unfortunately, due to it not being popular with the populace and its adaptability, it would take a considerable amount of time to learn. | Due to the wide range of development on Windows, it can be run on a wide variety of hardware, making it cost neutral, while also allowing it to be easy to learn s many tend to be familiar with it. | Mobile is a bit difficult to as while it does have a large base of tools to development, the requirements for the app to be run on different types of mobile, with differing specs makes it very difficult and expensive. |
| **Development Tools** | Swift, Objective-C, Xcode, Git, CocoaPods, Swift Package Manager (SPM), Sketch, Adobe XD, Figma, XCTest, Jenkins, Travis CI, GitHub Actions, Core Data, Realm, Homebrew, Visual Studio Code, Sublime Text, Atom. | C, C++, Python, Java, Go, Rust, Shell Scripting, GCC, Clang, Git, CMake, Make, GDB, Valgrind, Vim, Emacs, VSCode, Sublime Text, Qt, GTK, Docker, Jenkins, Travis CI, GitHub Actions, MySQL, PostgreSQL, SQLite. | C#, C++, .NET, Java, Python, Visual Studio, Git, MSBuild, Visual Studio Installer, WinDbg, Visual Studio Debugger, Visual Studio Code, Sublime Text, Atom, WPF, WinForms, UWP, Docker Desktop,  Jenkins, Travis CI GitHub Actions, Microsoft SQL Server, MySQL, SQLite, Chocolatey, NuGet, Powershell | There are many tools, some for android exclusively, some are for apple, some can be use to develop both.  Mainly both apps uses JavaScript as a foundation. |

## 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.>