

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 | 07/21/2024 | Zury Martinez | Prepare software design document, begin game application development, and address software requirements. |

**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)

Creative Technology Solutions (CTS) has taken on a new client, The Gaming Room. The Gaming Rooms wants to develop a web-based game that serves multiple platforms. Currently, their game, Draw It or Lose It, is only available in the Android store. The game consists of multiple teams each assigned multiple players. For functionality purposes, game and team names must be unique and only one instance of the game can exist in memory at any given time. This can be accomplished by creating unique identifiers for each instance of a game, team, or player.

## 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 following design constraints have been identified:

* The game needs to be accessible and functional across multiple platforms

The game is already available for Android OS, but we will still need to take existing code and tailor it to be functional for other OS’s

* Each team will need to be assigned multiple players

We will need to create code to distribute players evenly across teams.

* Unique Identifies for game instance, teams, and players

We will need to create code to check the uniqueness of teams and names. This will allow players and teams to know if the name chosen already exists in the system. We will also need to create unique IDs for games as only one instance of the game can exist in memory at any given time.

* Only one instance of the game can exist in memory at any given time

Each game will be given a unique ID

## [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 Gaming Room UML Diagram shows seven (7) classes. The main () method is contained in the ProgramDriver Class, which uses a Singleton Tester Class. The testSingleton () method checks to see if there is an instance of the GameService Class. The Entity Class is a Super Class for Game, Team, and Player Classes. These subclasses inherit from the Entity Class. The Entity class is a base class introduced to hold common attributes and behaviors. The GameService class has a zero-to-many relationship with the Game Class. The Game Class has a zero-to-many relationship with the Team Class. The Team Class has a zero-to-many relationship with the Player Class. Encapsulations used in the subclasses. In the diagram, you can see Team Class has a private list of players from the Player Class.

**"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** | MacOs is stable and consistent. Due to its consistency Mac users find it fairly easy to navigate. One of the downfalls is how expensive repairs and hardware is. Another disadvantage is the limited customization and closed ecosystem. | One of the biggest advantage Linux is the superior security protocols. Linux is also open source and offers cheaper and even free options. One of the disadvantages is that Linux is not user-friendly meaning its harder to navigate if you’re unfamiliar with said OS. | Window has a wide range of compatible software and apps along with extensive hardware for support on various devices. Window also has regular updates and security patched that allow for improved performance. A disadvantage of windows is that because it is a commonly used OS worldwide it is more prone to cyberattacks. | Mobile devices vary from device to device. However, mobile devices can connect to servers on other platforms.  Android offers several low-cost web server applications.  Most mobile devices are cloud based. |
| **Client Side** | MacOS is high in cost and is not accessible unless on another Apple device such as iOS or MacOS. MacOs offers some good security features. | Due to Linux being open source the OS is easier to maintain. Linux gives its user control over their system thus allowing users to customize how their system performs and looks. Due to these advantages a downfall could be security. | Windows is available in a wide range of prices making affordable to a large group. Window is also able to be customized. One of the disadvantages of windows is the frequent software updates and a if a user has many programs running simultaneously it could cause the Os to slow down and cause the user to reboot. | One of the biggest disadvantages is that a mobile device does not offer all the features that are offered and accessible via a computer. And one the of biggest advantages is the wide range, variety, price, and availability of mobile devices. Giving users access to different OS if they decide. |
| **Development Tools** | Relevant programming languages for MacOS are Objective-C, Swift, C++, Python, and Ruby. Some common tools used by developers is Xcode, Xcode Cloud, Visual Studio, and Pycharm. | Some of the better programming languages for Linux consist of Python, C++, Java, Rust, GO, and Bash. Some common IDE choices for developers are Visual Studio Code, Sublime Text, or Vim. | C++, JavaScript, SQL, Swift, and TypeScript are some programming languages used in Windows OS. One of the IDEs best known for Windows OS is Visual Studio. | Kotlin is a top programming language for Android. Some other options are C++ and Java.  Swift and Objective-C are the top 2 programming languages for iOS. Some of the more popular IDEs for mobile devices are VSCode, Eclipse, IntelliJ and IDEA. |

## 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**: Based on my readings and research, I would recommend using Windows OS. Windows can easily integrate the current Android application. Windows also offers many IDEs that can develop new cross-platform games and is one of the more popular OS worldwide which means there are more tools available to developers.
2. **Operating Systems Architectures**: Because of the Kernal processes does not affect the processes that allow the OS the function, I recommend using the hybrid architecture. With this, because one process does not affect the other, we can access memory, set up windows, and other processes simultaneously. The hybrid architecture also allows for customization based on need and improved security.
3. **Storage Management**: Since I have recommended Windows OS, I also recommend Microsoft Azure for storage management. Azure consistently updates and offers multiple storage options based on customer needs, Azure cloud, competitive prices, and customer service.
4. **Memory Management**: Memory management is already built into system utility. However, if needed we can consider other options such as One Drive.
5. **Distributed Systems and Networks**: a client-server distrusted system will be best to use here. This allows multiple players to connect to a single server for an optimized gaming experience. However, we must consider outages, network delays, scalability, and security. As this game’s success grows it is crucial that we also consider a strong server.
6. **Security**: Windows Defender is already built into Windows OS. But, I would also recommend implementing additional protocols such as routine audits and having a plan in place in case of a security breach which allows for a quick response time and reduced risk for the company, stakeholders, clients, and users.