

# **Draw It or Lost 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 | 03/19/2023 | Jesse Jesseman | Including the ability for a team play mode while allowing distinct usernames amongst different players. |

**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 would like to add multiple team game play with each playing having a unique online ID. Each team can have multiple players assigned at any given time. With that being said, only one instance of the game can exist in the memory at any given time. We can accommodate this by creating unique identifiers for each instance of a game, team, or player. The client would also like to expand to other operating platforms.

## 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)

* Ability to have multiple teams per game
* Each team member is allowed to provide input and be involved in the game
* Ability for the game to run on platforms other than just android
* Ability to check if the team name or player ID have already been chosen.
* There may only be one instance of the game at any given time.

The constraints listed above are what the game will need in order for it to run correctly. For the game only being run on android at the moment we would need to work with multiple developers so that the same game can be run on other operating systems and devices.

## [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 superclass in this model is the Entity class which is directly inherited by the Player, Team, and Games classes. The references like “id” and “name” will be shared by each class withing the diagram. The “SingletonTester” Class and the “ProgramDriver” class are what allows the multiple players from different teams to play one game at any given time from the list. Finally, “GameService” has a reference of “Games”, while “Games” has a reference of “Team”, and “Team” has a reference of “Player”. Another thing from this diagram is encapsulation, the class “Entity” protects certain data within the program by limiting the access to any public methods within said program.

**"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 servers are available on their website and can be relatively cheap to implement. It also allows for creating features easier than it would be on a widow’s server. With Mac servers, they only run a set number of devices which can limit your hardware possibilities. | Linux, which is built on Unix, is the most reliable and the easiest to use. The systems are also well known for the dependability of the servers. We will have to find some that is confident and familiar with linux. | Windows servers are more expensive than the alternative options but with it being one of the most used it will be easier to find someone who is skilled enough to provide our needs for this project. | Mobile devices are one of the last options as they do not have the power that computers do. This is the cheapest option but it is very hard to get a server started on the mobile side. |
| **Client Side** | Due to the need for specialized hardware and software it can get expensive, with that being said it is important to plan for these costs while budgeting for the development of the software. The time it takes to develop, test, and optimize any software for Mac OS is greater than that of the other operating systems. Finally the expertise for Mac OS is significantly harder to find as you need specialized knowledge when it comes to Xcode and the other apple-specific technology. | In the case of Linux clients, the cost may be lower compared to other platforms like Mac or Windows due to the availability of free and open-source development tools and libraries. The additional time it takes is due to the wide range of Linux distributions and hardware configurations available. The expertise needed includes the knowledge of the Linux kernel, shell scripting, package management | In the case of Windows clients, the cost may be higher compared to other platforms like Linux due to licensing fees associated with developing software for Windows. The additional time this may take is due to the wide range of Windows versions and hardware configurations available. The expertise needed may include knowledge of the Windows operating system, Microsoft Visual Studio development tools | Developing software that supports multiple mobile devices can be time-consuming due to the need to optimize and test the software for various operating systems, device models, and screen sizes. Developers must understand the unique constraints and capabilities of each platform and tailor their software development strategy to ensure optimal performance and user experience. |
| **Development Tools** | to build software for deploying on Mac, developers typically use programming languages like Swift or Objective-C, IDEs like Xcode or AppCode, and tools like Cocoapods, Homebrew, and Git. | The most common languages for linux include c/c++, Python, Java, and Ruby. The IDE’s and other tools include Eclipse, Visual Studio Code, and Git. | The most common languages on windows include C#, Visual studio, .NET, and Microsoft SQL server. Other popular tools and frameworks for developing Windows applications include Xamarin, Unity, and Electron. | The most common languages include Java for android, Swift for iOS, React Native and Flutter both for cross platform. Some IDE’s and tools include Visual studio, PhoneGap, and unity. |

## 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**: My recommendation would be for The Gaming Room to start with windows devices as you will not run into a problem such as a shortage of IDE’s to work on. The cost will also be less than that of others and the expertise needed will also be lower.
2. **Operating Systems Architectures**: Windows offers a range of services that are utilized by all Windows-based applications. These services empower applications to display a GUI and access system resources, among other functionalities such as Graphics and Multimedia, messaging, and web services. These services can be accessed either through a user account or a dedicated server.
3. **Storage Management**: One of the useful features that comes with Windows is storage sense. This feature enables you to examine and manage the files on your hard drive, including their size. Additionally, it provides the convenience of designating specific locations to save apps, making them easily accessible. In addition, similar to other devices, you can also use cloud storage to save data. With the built-in storage system, creating and organizing files for significant projects is effortless, preventing the risk of misplacing or unintentionally deleting them.
4. **Memory Management**: As you embark on the game development process, it's crucial to establish a database or library containing numerous images. Utilizing the memory allocation feature facilitates seamless storage of pictures beyond the default picture folder. Consequently, this approach enables you to maintain your entire project in a more secure area on your computer. Furthermore, you can access and utilize the pictures conveniently while working with the Integrated Development Environment (IDE) and opening files to create the game.
5. **Distributed Systems and Networks**: Due to the differences between each operating system, I explored options to make the game compatible with all devices. After some research, I discovered Develop 4, an Integrated Development Environment (IDE) that facilitates cross-platform game creation. This IDE can be utilized on any device, and once the game is created, it can be exported to various platforms such as the web, iOS, Android, and more, facilitating cross-play. It's vital for the company to ensure that their servers are robust enough to support a large number of players, along with backup power to deal with power outages.
6. **Security**: Windows 10 offers several built-in security features such as Windows Defender Antivirus, firewall, and automatic updates. Other operating platforms, such as macOS and Linux, also provide robust security measures, including encryption and malware detection. Windows comes with integrated security protection software, although it's advisable to utilize additional security measures to safeguard user data and information. However, concerning the system's default features, Windows comes with pre-installed protection that detects malware, viruses, and security threats in real-time. To ensure the security of the system and user information, the system receives automatic updates to address evolving threats.