

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 | 04/20/24 | Nicholas Zunno | Initial prototype for software design |

**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 development from an app-based game to a web-based game “Draw It or Lose It”. This game will be designed for The Gamming Room by the consultant at Creative Technology Solutions, (CTS). The game is currently only offered as an app on Android but is going to be opened to multi-platform. CTS is providing the technical support, and make sure the project will be developed to meet the clients five key software requirements. Draw It or Lose It will be able to host one or more teams of unique identities with multiple players per team. This project must meet the industry standards and to enure the project is scalable and bug free and functions properly

## Requirements

The need to continue on Android add IOS and the three software platforms, MAC, Windows,& Linux

The game need the team names to be unique

To alert the team captain if a team name exists and have them choose another name.

## [Design Constraints](#_2et92p0)

1. Storage:

Meeting the clients need for storage capacity and collection of stock photos.

1. Security:

The system need to maintain security of login and be able to differentiate between different players and authorizations.

1. Individualized logins:

No duplicate login id’s

1. Programing language:

The current language is for android in java.

1. Team name, number of games , and players name and quantity:

Need team management system for unique team names and player autorizations

## [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 ProgramDriver Class is the main method of play. ProgramDriver uses SingletonTester to test of there is an instance of GameService already. The Entity class is the parent call of the game, team and player classes. Each game, team, and player inherit the required attributes from the Entity class. A Player can’t have a Team, but a Team can have a Player. A Team cannot have a Game, but a Game can have a Team. A Game can’t have a GameService but a GameService can have a game. GameService ca only have an instance of each game running at any given time. Each game can only host one unique team and one team can only host one player at a time

**"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 utilizes flexible terminal commands when configuring access, the server, or when making changes.  Characteristics:  It is highly used web browser  **Advantages**;  mac upgradeable, and has multiple options for different web browsing requirements  **Disadvantages:**  **Mac** less less popular for web hosting services | Linux is similar to Mac’s flexibility with terminals, but at a much lower price.  Characteristics:  Secured and the most open sever.  Advantages;  Security flaws are caught very quickly, and it as the most preferred web browsing host.  Disadvantages:  It is the most difficult server to find support for web hosting | Windows as the most software available when compared to all the other OS.  Characteristics:  Windows is the most dominate compared to others platforms and is closed.  Advantages:  Windows has more compatibility, and high recourses requirement make for faster load times.  Disadvantages:  Susceptible to virus’s and has little tech support. | Mobile devices work better if the server mmobile and if it can be trafficked to one place.  Characteristics:  The most popular and very versatile and portable, and its the least expensive  Advantages:  Mobil has the largest reach, with the best combatability.  Disadvantages:  Very selective in mobile devices, and has weak security. |
| **Client Side** | Midlevel expertise and time is required, with the cost being simialr to Windows. | This require the most experienced and time but is the lowest cost. | Pretty much same as Mac in the midlevel requirements, with similar costs. | mobile devices provide the most flexibility to clients With upgrades, it is also slightly more difficult to implement than other mobil devies |
| **Development Tools** | While running different languages in Mac can swift the most popular tool. Macs languages consists of HTML/CSS/JavaScript | Linus can be utilized with visual studio, eclipse, and notepad++ which makes it nice and easy. Utilizing vast languages as tools, such as | Visual Studio Code is the best way to code using Windows applications. You can Utilize almost any language of C++ to program in Windows | You need to develop for android, which requires a person specialized in Android studio. IOS needs a MacBook with iCode. Or using C++ some can develop a code that can be converted to either IOS or Android. However a MacBook is still required to convert C++ to IOS. |

## 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 would recommend using Linux to host the game. Linux has an open source and that can reduce the cost, plus it support multiple development tools.
2. **Operating Systems Architectures**: Linux has a very stable and secure system.
3. **Storage Management**: seeing how Draw It or Lose It is a web-based game I recommend using a cloud based storage system. Using cloud storage, we can grow as needed and don’t have to over invest in the initial memory.
4. **Memory Management**: Linux is a very easy and versatile software that’s easy to customize. So for this game I would use Java as it can be converted to other operating systems as needed. Jav also does memory management automatically using a garbage collector, therefore no need to implement a memory management.
5. **Distributed Systems and Networks**: From my evaluations google chrome browser is available on on all platforms. Since the game is web-based anyone on any OS with a google chrome browser can access and play.
6. **Security**: using google cloud services we should have to worry about data center security as google provides the hardware and infrustrusture.