

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template**](#_l6ti7uoag22u)1

[**Table of Contents**](#_30j0zll)2

[**Document Revision History**](#_grjogdjh5fi8)2

[**Executive Summary**](#_sbfa50wo7nsh)3

[**Design Constraints**](#_2et92p0)3

[**System Architecture View**](#_ilbxbyevv6b6)3

[**Domain Model**](#_8h2ehzxfam4o)3

[**Evaluation**](#_2o15spng8stw)3

[**Recommendations**](#_m8aleynsvzvc)5

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.2 | 10/13/2022 | LaNise Essick | * Teams and players can now be created and added * Ensured that a player or team name could only be used once. |

**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 Gaming Room is a client that wants to extend their already developed game to serve multiple platforms. The game is titled " Draw It or Lose It". Currently, the game is only available on the Google Play store for Android phones. Expanding the game to multiple platforms will increase the revenue and customer base of the application.

## [Design Constraints](#_2et92p0)

* Each game must have teams, whether it be one or more.
* Each team will include multiple players.
* To ensure that users do not confuse one team or game for another, names must be unique.
* There should never be more than one instance of the game running.
* Must be compatible with multiple operating systems.
* Language Java
* Framework API

In order for the game to run properly, the developers must adhere to these design constraints. It is also important to the client that the game runs on a variety of platforms. However, it is using code that is only compatible with Android devices. The code must be refactored / redesigned to run on other platforms, which requires more development time. There could also be a problem with cloud space when developing the web application or a limit on the number of games, players, or teams due to server limitations.

## [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)

There are seven classes in this UML diagram. This example of multiple inheritance shows that Game, Team, and Player are generalization relationships that reference the same general class Entity. Our three classes, along with the GameService class, share a direct relationship and multiplicity, indicated by the curly brackets that say our classes might share zero to many objects. There can be more than one player on a team, and each player has a unique id and name. It is possible to add more than one team to a game, each of which has its own name. GamerService contains a list of games as well as a singleton method called service. This class also has references to the game class. Furthermore, we have a class called SingletonTester that tests whether the ProgramDriver class is running only once at a time. The main method for the Terms of Use is stored within the ProgramDriver 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** | In Mac OSX, you can configure the server and what can access it with more flexibility in the terminal. However, you are limited to Apple-only hardware options. | In terms of reliability and stability, this is the best option. A variety of hardware options make it very cost-effective. | Several hardware platforms are supported with a variety of release options. Additionally, HTML files are supported. | Long-term storage and coding of this information would be difficult. The best option would be to look at other options. |
| **Client Side** | It is expensive and the end user must buy a product developed by Apple. The software is less versatile unless it is loaded into another operating system using a virtual machine. | Even if it were cost-effective, most people would have a difficult time learning and using Linux given that they are not native to it. Despite being available on a variety of machines. | Software is most readily available and open. The majority of users have some familiarity with Windows, so the ease of use is minimal. Various machines are also supported. | Battery life and form factor are limited. However, clients and users will receive ready updates. |
| **Development Tools** | The primary programming language on Mac OS is Objective-C. | This tool supports most programming languages: C, C++, CSS, Java, JavaScript, HTML, PHP, Perl, Python, Ruby, and Vala. | Besides C, C++, CSS, Java, JavaScript, HTML, PHP, Perl, Python, Ruby, and Vala, this program supports most other programming languages. | iOS developers typically use Swiftic, while Android developers typically use Java. |

## 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 recommend the Gaming Room to start their project with the Windows operating system, because it is easier to use than other alternatives. For gaming, Windows is the way to go. The cost for Windows OS is relatively low and it comes with helpful programming tools that make coding faster.
2. **Operating Systems Architectures**: The layered design of Windows consists of two primary components: user-mode and kernel mode. Services—including file management and graphical user interface (GUI)—are used by all applications, including those developed for Windows.
3. **Storage Management**: Because the image library can be accessed through a cloud storage solution, and because it does not take up much space, adding to or modifying the existing images is quick and easy. The Windows server operating system allows you to manage memory by allocating space on your hard drive and deciding what files are stored there. The ability to save files and applications in a specific location makes it easier to find them, while using the cloud helps with load times.
4. **Memory Management**: Windows OS has its own physical address space and a virtual one that enables addressing of memory. Each process has its own set of virtual addresses, but threads cannot access memory that belongs to another process—this protects one process from being corrupted by another. A 64-bit architecture will allow for more virtual memory, speeding access to image files.
5. **Distributed Systems and Networks**: Using a cross-platform development tool such as Unity or Unreal Engine 4 would allow you to develop for all platforms regardless of the operating system. I'd recommend using the Unity game engine because it's cost-effective and supports Windows, Linux, Android, and iOS. A dedicated gaming server would make a great choice for handling high traffic levels—and making sure your application is always online to give players a seamless experience.
6. **Security**: The user account control settings in a Windows server operating system can help protect data going into and out of the system. The operating system assigns each process a unique set of computer instructions known as virtual addresses. These can only be accessed by the process they were assigned to, not by any other processes on your machine. A VPN's ability to protect a user's data and accounts from being compromised by third parties is its most important feature. Windows comes with anti-spyware software built in, but other programs such as Mcfee and Norton can be purchased to help protect the system.