

Draw is 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 | 09/19/24 | Zackery Spears | <Brief description of changes in this revision> |
| 1.1 | 10/06/24 | Zackery Spears |  |
| 1.3 | 10/18/24 | Zackery Spears |  |

**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 company Gaming Room has a popular Android app called Draw It or Lose It. Gaming Room wants to develop a web-based game that serves multiple platforms based on the current game. The game consists of various games that host numerous teams, each with multiple players. Each game and team name must be unique for the game to function correctly, with four rounds that last a minute each. Draw It or Lose It contains an extensive library of images that each team has to guess what the image is for the time runs out. The company at The Gaming Room does not know how to set up the environment.

## 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)

* Must be able to run on Android, IOS, and the web.
* Each team should have multiple players.
* The game must only have one instance at any time.
* Most have unique ID’s for each instance of a game, team, and player.
* Game and team names must be unique to allow users to check whether a name is in use when choosing a team name.

## [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, uses Directed Association with SingletonTester to verify the existence of an instance of GameService. The Entity class is the parent class to the Game, Team, and Player classes, with the latter three inheriting the Entity's required attributes. It's important to understand that a player cannot be a member of a team, but a team can have a player. Similarly, a team cannot be associated with a game, but a game can have a team. A game cannot be linked to a game service, but a game service can manage a game. These constraints are crucial as they ensure that the game service handles only one instance of each game running at any time, and each game can have only one unique team and one unique player at any 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 has expansive server hosting. Mac servers are easy to configure. Has very simple interface. | The licensing is super cheap. Difficult to navigate the platform do to everything must be run through the terminal. Uses  command shell  for simple server  config and access. | Friendly user interface. Easy to setup servers. The licensing expansive. | Mobile devices lack the power to run high end servers. Oracle is on of the  server-side  companies  available. |
| **Client Side** | Mac might be good platform but it is super expansive. Apple products cost a lot and to develop in Swift everyone will need to own a Mac Book. | For the client side, a lot of time and knowledge required. Must use a Linux operating system to access the date. | Easy to learn and understand do to all the windows support. Windows is more expansive than Linux. | Provide flexibility to clients or even developers to see updates at any  place. Slightly more difficult to implement than other devices |
| **Development Tools** | Swift for IOS development.  HTML,  CSS and JavaScript for Web Development, Java  Visual Studio Code or Eclipse as the IDE | HTML,  CSS, Java and JavaScript for Web Development  Ultimate IDE | HTML,  CSS, Java and JavaScript for Web Development  Visual Studio Code or Eclipse as the IDE | Development for Android devices developers use Kotlin and Java as the program language and uses Android Studio as the IDE. A developer must have a MacBook and use Swift to develop the app for iPhones. |

## 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**: The most appropriate operating platform for The Gaming Room would be Windows. The operating system provides the most user-family system while providing tones of support and different IDE. Windows also would work with the current Android build, allowing developers to design, develop, and deploy apps and solutions for Windows PCs easily using a suite of Microsoft products.
2. **Operating Systems Architectures**: Microsoft Windows is a graphical operating system developed. This operating system allows for software, games, files, videos, coding, and connecting to the internet. Windows also divides its operating system into two distinct modes: user mode and Kernel mode. User mode processes are those that users directly interact with, influencing the user experience significantly. In contrast, Kernel mode operates at a lower level, managing essential functions such as input and output operations, memory management, networking, hardware control, and system routines. Additionally, Windows employs a directory structure for data organization. The operating system also accommodates multiprocessing and hardware modularity, enabling users to customize their systems effectively.
3. **Storage Management**: Storage sense is how Windows 10 and 11 calculate how much storage is being used and manage files on your system. This allows you to manage files on your hard drive and even helps the user by automatically free up space on the drive. This is used when clutter is no longer needed such as temp files.
4. **Memory Management**: Window 10 and 11 storage sense would allow for the game Draw It or Lose multiple photos and game players in one secure place. The OS also uses a technique called Swapping which allows programs to swap in and swap out between the hard disk to the RAM.
5. **Distributed Systems and Networks**: Multi-interaction systems that are network-based usually have to come up with original designs for player communications and data sharing. While using HTTP, you can load webpages using hyperlinks. This allows information to transfer between clients and servers.
6. **Security**: Windows comes with built-in security protection software like firewalls and antivirus software. However, to secure user data and information, it is recommended that another source be used.