

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 3.0

## 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 | 03/19/25 | Jimmie II Garrido | Filled in Executive Summary, Design Constraints, and evaluated the Domain Model. |
| 2.0 | 04/4/25 | Jimmie II Garrido | Filled the Evaluation table that includes the categories service side, client side, and developmental tools. |
| 3.0 | 04/18/25 | Jimmie II Garrido | Filled the recommendations sections that includes the OS recommended for the client. |

**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)

We are working with a new client, The Gaming Room, which have created their current game Draw It or Lose it. The game is about teams competing against each other and there are rounds where both teams have a chance to guess what the images are. Each game has four rounds with a time limit of one minute each. They want to expand their game by making it web-based which will serve multiple platforms. It is currently available on the Android App only, now we can discuss how to implement the game’s cross platform compatibility.

## 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)

A design constraint we may be facing for this game application is cross-platform compatibility. As addressed in the executive summary, the game is currently available only in the Android app only. We need to find ways to make this app compatible with other platforms and it performs consistently in their respective platforms.

Another design constraint would be the user interface (UI). The game has been developed for a mobile app Android. If we plan to implement the game onto another platform such as Windows. We need to decide whether we keep its current UI on the Android platform or tear it down and recreate similar concepts for the Android and the rest of the platforms.

One of the requirements specifically requested by the client is a design constraint and it would be the unique player/user Id’s in the games and teams. We need to create a system in the application that doesn’t take any existing names and duplicate matches. There needs to be unique names that aren’t in existing teams so we can keep track of the different games happening throughout all platforms. There needs to be unique identifiers for each of the game, teams, and players.

Another design constraint would be the game application following the standards of the different platform guidelines and restrictions. We need to make sure that this game adheres to the policies to each of the different platforms so they can be released with no issues.

## [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)

In this UML diagram we can see all the different classes that are in this package. We can see on the top left there is the Program driver that contains the main(). You can see that it connects to the Singleton tester. The program driver will use the singleton tester to find any instance in the Game service class. Next, we can see the Entity parent class and it displays inheritance relationship with the player, team, and game class which they inherit information. You can see the Game service class has a relationship with the game, team, and player classes that show a “0…\*” link between each other. As the requirement requested from the client, this will create a structure of an instance of a game, team, and player class. This will make sure that it will iterate throughout all the Ids created in the lists and ensure there is one instance of the Game service.

**"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 is developer friendly and has a robust and customizable environment for the managing servers. It also integrates all other Apple devices which makes it more convenient for the various devices. MacOS can be used as a server but a major weakness in development using Mac is the price which is very expensive. | Linux has a reputation of being a very secure OS which helps prevent any cyber-attacks. A great advantage with this OS is the stability it offers; it has minimal downtime and great performance. One major weakness would be its lack of user friendliness. It takes a deeper learning curve since it would take more time to adapt to the systems tools and commands. | Windows is another user-friendly OS, which has simple management and administration tasks in the UI. There are regular updates on this OS regarding the system improvements and security updates. Some weaknesses would be the security, it is known to be prone to malware. Another would be licensing fees for hosting web-based applications. | Mobiles devices can have web applications be accessed on web browsers. They are also highly portable so you can access the web apps almost anywhere you go. Also, you can use cellular or Wi-fi which also could make it slow to run the applications.  Some weaknesses we should shed some light on are how it will have somewhat poor performance in comparison to other platforms. This could lead to crashes and become difficult to use. |
| **Client Side** | The cost would be very expensive for users. Mac devices are more on the costly side. The time can be consuming for testing and development in the MacOS. The users need to have a decent amount of experience to navigate the OS to be able to maintain the software over time. | Although some of the features are free in this OS, we could potentially need to spend for specialized tools needed for specific projects. It's not really user friendly so we’ll need to train their knowledge and experience in using LinuxOS which could cost us some time. May need to organize and develop a team of developers to train and learn the system for efficiency. | WindowsOS would be more costly compared to Linux having to deal with licensing fees on running servers. This OS is user friendly so it’s easier to find people with expertise in using windows. The updates and maintenance could be time consuming. | The cost of mobile devices is generally cheaper than computers. You can use smartphones or tablets in considering this route. Using mobile devices can be very flexible for all users. The time could be very consuming in developing apps since the hardware is slower in comparison to other platforms. Using mobile apps would be more difficult since we use our hands and type on the screen. I could suggest using external keyboards for fast typing. |
| **Development Tools** | Some programming languages used on Mac are JavaScript, CSS, HTML. Some tools used are like PyCharm, WebStorm, Visual Studio, etc.. | Programming languages used on Linux are Java, JavaScript, Python, PHP, etc… The tools used are PyCharm, Visual Studio, Eclipse, etc… | Some languages used on Windows are JavaScript, HTML, CSS, Python, Java, C#, etc… The tools being used are GitHub, PyCharm, Visual Studio, etc… | Some languages used on Mobile Devices are CSS, HTML, JavaScript, Java, Python, etc... Since it was created on Android the tools are Android Studio. |

## 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**: An appropriate operating platform that I would like to recommend is LinuxOS for expanding The Gaming Room’s game, Draw It or Lose It.
2. **Operating Systems Architectures**: The architectures of this OS would be a layered structure following with hardware, kernel, and shell. The hardware includes physical components such as the CPU, RAM, storage devices, and input/output devices. The kernel is the OS and handles the system processes and apps. It controls the overall systems operations in running the OS such as processing, memory, system, etc. The shell is the command interpreter for the user and kernel by entering commands
3. **Storage Management**: I recommend pCloud for storage management in this OS. pCloud is affordable and it does contain strong security features. We can expand the storage and its lifetime storage. It has fast syncing and upload speeds so it’ll be easier to navigate between all operating systems.
4. **Memory Management**: Some great memory management techniques included from this OS would be virtual memory. This memory management technique benefits the Draw It or Lose It software by allowing the application to function as if they have large consistent memory space, regardless of the amount of RAM. This technique is great to be deployed on multiple platforms and it optimizes the number of resources being used.
5. **Distributed Systems and Networks**: One thing we need to consider is the possibility of an outage. The web-based application will then go offline and become inaccessible. We would need to consider a team that will handle the systems and networks and find the root of the cause. Since it is a web-based application we can connect to many other devices and reboot the servers to prevent downtime. If we are unable to do so, we can contact our social media team to post approximate downtimes to keep our community posted. We can then learn from this outage and find ways to prevent another outage or find faster ways to reboot the servers for all our users to access the application.
6. **Security**: An important factor to consider for this web-based application would be for the security and protection of all users and developers for this web-based application. We can consider using CloudLinux OS since it provides many security features that’ll ease the users. It offers CageFS that will virtualize user file systems which limit access. It also includes SecureLink which is designed to prevent malicious attacks and HardenedPHP. Since it's a web-based application we can also consider a Web Application Firewall that’ll scan and filter the HTTP traffic and protect the web application from injection attacks, XSS, or cookie poisoning. With these security features offered in this OS, this can help ease the users’ mind knowing that security features are placed across all platforms.