

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 | 3/23/25 | Robert Szabo | Initial |

**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 new client that is seeking to develop a web-based game called “Draw It or Lose it” loosely based upon the game show “Win, Lose, or Draw.” They require our assistance to set up the game environment and we will need to streamline the development.

## 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)

* Each game must be capable of hosting one or more teams
* Teams must be capable of consisting of multiple players
* Game names and Team names must be unique to allow users to determine if names are already in use
* Only one instance of a game can exist in memory at a given time

The implications of these constraints on development are that our code must reflect the ability for each team and game to have a unique identifier allowing each game/team combination to remain unique in each instance of the game. Additionally, each player ID must be unique so that each individual player can be identified by the instances of the game and team and only able to be in one game and on one team at a time.

## [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)

This UML class diagram illustrates the use of seven classes. The entity class serves as a parent class to the Game, Team, and Player classes, and those classes share a generalization (inheritance/”is-a”) relationship because each on exists as an entity within the program. All three of these classes, with the GameService class, share a direct association and represent a “zero-to-many” multiplicity, meaning the association flows in one direction and is not reversible and that an instance of the classes from right to left (Player to GameService) can be associated with zero, one, or many instances of that class at the other end. In the diagram below, you can see that represented by how the Player class has instances in the Team, Game, and GameService classes, and how that relationship does not flow in the opposite direction. Additionally, we have a SingletonTester class that tests if one instance of the game is running at a given time associated with the ProgramDriver class. In the ProgramDriver class itself, this holds our main() method which executes the methods instantiated by the other classes within the “Draw It or Lose It” 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.

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Mac is known for being stable, reliable, and secure. Macs have access to iCloud services to offer extra storage and easy data access. It is also know for being easy to learn and use. Mac OS is highly resistant to malware and viruses due to a Unix-based operating system as well as because it has fewer users than the more popular Windows Server. Mac also has support for web servers like Apache, PHP, and NGINX. Despite these advantages, Mac servers are usually more expensive than others, but due to their stability and reliability, you may save costs down the line in technical support and maintenance costs. | Linux is Unix-based and open source meaning security and reliability are some of its strengths. It is also known for being cheaper relative to the other major players due to its open-source nature. The Linux OS is free to use, so fees that you would have to normally pay for licensing in an enterprise environment are no longer a factor. Also, because it is open source and updated frequently with distributions releasing every few months, businesses can save money on technical support because these free updates often solve recurring problems. Linux is also known for a very high level of security due to advanced security features that are constantly put under scrutiny by the large number of developers working on this open-source OS. This means security updates are frequent in order to address threats that come up as quickly as possible. It is also known for being very scalable, working just as efficiently on a small business’ server as it would on a multinational corporation’s. Its major downside is that it is not as user friendly as Windows or Mac, meaning more time and money will need to be dedicated to training and development for employees become and remain proficient. | Windows is best known for being compatible with many kinds of software. This ability to integrate with development tools and applications can make managing the server easier and improve productivity. Offers OneDrive cloud storage option to offer extra storage and easy access to data. Windows Server falls in the middle of Mac and Linux in ease of use; where Mac is simple, some users may have a difficult time learning to use this software due to a more complex UI that offers a higher number of settings and options than Mac. One downside is that Windows Server is potentially more vulnerable to cyberattacks because of how popular the platform is. This popularity means attackers spend more time and resources dedicated to this server system than on others. | While a mobile device can be used as a server, their capabilities pale in comparison to Mac, Linux, and Windows Servers because this is not their primary function. |
| **Client Side** | Users on Mac devices come preloaded with the Safari web browser. This browser is capable of reading JavaScript, CSS, and HTML. JavaScript is very common in web development across multiple client types as well as CSS. HTML is global standard for markup languages for web pages. But, as stated earlier, Mac remains the most expensive option to develop on. | Since Linux is a bit more “hands-on” than other operating systems, most of the time users will install common browsers from the web such as Google Chrome or Firefox to access web applications. Like Mac, these browsers are compatible with JavaScript, CSS, and HTML. This OS is great for experienced software and web developers because it is so cost-effective, but it has a barrier to entry for many that do not have experience with it because it is more difficult to learn than Mac or Windows. | Windows devices now come with a web browser called Microsoft Edge (after phasing out Internet Explorer) which is built on the Chromium open-source project, making it very similar to Google Chrome. Because of its similarity to Chrome, it reads the same languages, including the most common JavaScript, CSS, and HTML. Windows makes a great development option because most companies already use Windows devices so using tools for this OS will remain the cheapest option. | Cell phones have a UI that is unique compared to traditional computers. Most use touch screens and run on special operating systems like Apple’s iOS or Linux’ Android OS. Windows hasn’t generally done well in the mobile market. A great benefit of developing here is that just about everyone has a smartphone nowadays, making accessibility very high by potential users. However, the differences will require significant UI design changes to ensure they properly format to a smartphone. Despite this, I believe that due to the incredibly high value proposition in the smart phone market (both iOS and Android), developing for clients in this space is essential for business success in this venture. |
| **Development Tools** | Apple’s official IDE is Xcode, but you can also use other IDE’s like VS Code or even AppCode (specifically designed for macOS and iOS development). You are also able to use Git, a popular open-source version control system, which is key in allowing teams of developers to collaborate efficiently on concurrent projects. As far as programming languages go, some of the most popular include Swift (the preferred language for Mac/iOS), C++ for system programming, Python, and JavaScript. | Since Linux is open-source, it does not have official tools, however some are designed around and commonly used in the Linux environment. These include Eclipse IDE, and PyCharm/IntilliJ IDEA/Webstorm for Python, Java, and web development, respectively. VS Code is also a viable option. As with Mac, Git is the preferred version control system. For languages, C and C++ are very popular for system programming and Python is popular for scripting and other applications. For web apps and server-side development, Java is preferred. | Visual Studio, Microsoft’s own IDE, is preferred by many developers using Windows, but due to Windows’ flexibility, many other IDE’s can be used such as PyCharm, Eclipse, or NetBeans. For cloud applications, the Azure suite of tools can be very useful. Preferred languages include, C# and C++, as well as Python or Java for some desktop applications, and JavaScript for web development. | Native IDE’s for mobile devices include Android Studio for Android and Xcode for Apple iOS. Other cross-platform tools include Flutter, React Native, or Ionic. For iOS devices, a key language is Swift. Additionally, Kotlin and Java are important languages for Android application development. You can also use languages such as Javascript (using React Native for cross-platform support) and Dart (with Flutter) to build cross-platform applications. |

In each cell, remove the bracketed prompt and write your own paragraph response covering the indicated information.

## 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**: <Recommend an appropriate operating platform that will allow The Gaming Room to expand Draw It or Lose It to other computing environments.>

I would recommend using Linux Server as the operating platform for this application. This is because Linux is, for one, free to use, and with an application that is very simple and just starting out, you probably won’t need to invest too much capital into it to start out. It is also a good choice because Linux, when compared to its peers, is known for being very secure due to its regular and frequent updates in an open-source development environment. Another important benefit is that, for an application development company with experienced software developers, Linux is highly customizable. Developers can access code repositories, customize the kernel version, and even modify the source code to meet your needs. Additionally, the front end can be agnostic to the back end and communicate through API’s, meaning you can write front end code in whatever language is best for the end user device platform and it can easily communicate with the back-end architecture.

1. **Operating Systems Architectures**: <Describe the details of the chosen operating platform architectures.>

The Linux server architecture is comprised of a layered structure that includes the kernel, which is the core of the OS, interacting with all the hardware of the machine, the shell, which is the command-line interface, and the system library, which provides functionality for applications and are critical for managing system resources.

In the software development context, this backend server will be able to manage the core environment of the game while the front-end functionality can be handled on the client side. Choosing to use the front end to handle the functionality of the game can reduce the load on resource-intensive aspects of the application, requiring less expense on server maintenance.

1. **Storage Management**: <Identify an appropriate storage management system to be used with the recommended operating platform.>

For storage management, I recommend using MEGA because, for one, it supports Linux natively, so it should not have any compatibility issues with this OS. Additionally, MEGA is incredibly price flexible, offering a tool to tailor your storage quantity to exactly what you need, priced on a sliding basis depending on how much storage you need. It also includes a VPN service and MEGA Pass, a password service with multiple layers of security to prevent anyone from accessing passwords. They boast that they have never had a data breach in over a decade of operations. All in all, MEGA offers an incredibly secure, encrypted storage option that is optimized for use with Linux and I definitely recommend cloud storage for this project.

1. **Memory Management**: <Explain how the recommended operating platform uses memory management techniques for the Draw It or Lose It software.>

A Linux server manages memory for application hosting by using a few different methods. For one, it employs virtual memory, allowing applications to access more memory than physically available on the server if need be, by using RAM and swap space to “create” more RAM as needed. Swap space is a technique used in memory management when you use a portion of the hard drive’s storage as an extension of RAM.

Linux server also uses a technique called page caching, where the OS stores data from the user’s web requests in a cache so that when the user requests the same data again, it can check the cache, and if found, serve the data to the client without needing to re-fetch the data from the server. This causes faster load times for the client.

Additionally, Linux can use a process called overcommitment, where the kernel can allocate virtual memory without first allocating physical memory. Doing this can make the server appear to have more memory than it does in reality, allowing more processes to run concurrently. By default, the system allows overcommit but refuses memory requests that are too large because there are issues that can arise when a process attempts to access memory that the system does not currently have the ability to provide, resulting in the kernel killing other processes to free up memory, leading to potential system instability or problems with performance.

1. **Distributed Systems and Networks**: <Knowing that the client would like Draw It or Lose It to communicate between various platforms, explain how this may be accomplished with distributed software and the network that connects the devices. Consider the dependencies between the components within the distributed systems and networks (connectivity, outages, and so on).>

Because we will be using cloud-based storage, many issues such as unexpected downtime and outages can be avoided because the cloud providers are able to make adjustments to their services to avoid big outages.

With the front-end being rendered on the client-side and the backend operating on the server side, we will make them communicate through the use of RESTful API’s. Doing so will allow the client and the server to communicate clearly to the end user’s OS.

1. **Security**: <Security is a must-have for the client. Explain how to protect user information on and between various platforms. Consider the user protection and security capabilities of the recommended operating platform.>

The concept of the least privilege will be utilized in our application, meaning that no user will have access above exactly what they need to perform their function for the application. This means that end users will be limited to low-level access such as entering a lobby to find other players to join their team, sending friend requests, directly inviting other users to join their teams, and creating private and public matches.

All accounts will be password protected with the use of MEGA’s service that generates random passwords as well, should our users desire complex passwords that are difficult for a machine to crack.

End users will not be allowed administrative access to the system, and administrative access will still be tiered based on our employees’ roles, i.e. developers can edit code, business managers can access and view code, etc. Our API’s will also be protected through encryption, and we will add a firewall to protect the server as well.

Additionally, we will make sure maintain a strict schedule to update and patch our servers regularly based on Linux’ regular updates to ensure system security remains up to date.

References

Ha, D. (2025, March 10). *9 reasons linux is a popular choice for servers*. LogicMonitor.

ht[tps://www.logicmonitor.com/blog/9-reasons-linux-is-a-popular-choice-for- se](tps://www.logicmonitor.com/blog/9-reasons-linux-is-a-popular-choice-for-se)rvers#:~:text=Open%2Dsource%20nature%20and%20customization,system%20runs%20efficiently%20and%20effectively.