

Draw it or Lose it 2

# **CS 230 Project Software Design Template**

Version 1.2

## 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 |
| --- | --- | --- | --- |
| 1.2 | 04/19/25 | Antonio Colantuono | Added a Recommendations section |

**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 wants to develop a web-based game that serves multiple platforms based on their current game. The current game is currently only available on Android. The company does not know how to develop the environment and requires assistance with streamlining the development while adhering to their software requirements and also maintaining a cloud server to host the game on.

## 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)

1. **Hardware and Software Compatibility:** The company must be able to afford a server to hold the game servers since it is an online only game. The client also needs to look into leasing software that can be used to develop the app on a multitude of different platforms and be able to do it almost bug free. The client can also look into out sourcing the application development for the game to run on different devices to meet deadlines.
2. **Budget Constraints:** Purchasing the software to code with is expensive so the client can look into either outsourcing the development on specific devices or leasing the software and spending the money to train employees on how to use it.
3. **Staff Training & IT Support:** If they choose to lease software, staff must be trained to use the new system, and IT must be prepared to support a new operating system.

## [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 UML diagram has everything related to the package com.gamingroom. Programdriver class holds the main function while game service is inherited by game which is then inherited by team and then player which then all of them are used in one form or another by the Entity class. Since the arrow at the Entity class is see through, we can assume that it inherited by everything below it. The Entity class is at the top of the diagram representing that it inherits everything below it and the way the UML diagram is structed is showing polymorphism since Entity can represent any of the classes it is inheriting.

**"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 regarded as one of the most user-friendly software systems out there due to how easy it is for anyone to use on their computers, mobile phones and tablets. Since it is based on Unix it is known to be more reliable and better suited for critical applications and server management. MacOS also has a built-in web server which can be helpful when it comes to testing and updating any issues on the server side. Mac also has easy keyboard inputs and shortcuts making development easier and less strenuous. On the cons however, macOS is not as well trusted when it comes to security for servers as something such as Linux which can lead to issues if there are high server loads. Since not many people build servers in macOS there is not as many outlets if you have bugs in your server’s vs building in Linux which is more well known and used for server management. Running a server on macOS is also very intensive resource wise which could affect the performance of the server and performance of the overall mac the server is running on if that mac is needed for other tasks. | Linux is open source which means that there are no licensing fees that are seen on other operating systems and it is more well used than macOS leading to easier to find solutions for bug fixes since there is a higher chance someone on the internet has had that same bug happen before. Linux also has one of the most top-notch security features and capabilities meaning that you are less likely to be attacked by cybercriminals or DDOS attacks since it is harder to hack into. Linux servers are also able to run for long periods of time without rebooting (most games use Linux servers: Minecraft, Among US, CS-GO etc.). Since Linux is open source, it also features a wide range of different programming languages you can use to build and update your servers. However, since Linux is so well known and advanced it can be hard to learn at first usually requiring a lot of self-education time on their servers. You usually need a deeper understanding of how servers work and operate than you do with macOS or Windows since those are easier to work with and more user friendly. Since it requires a deeper understanding to use this can lead to longer times for bug fixing, server updates, client fixes etc. | Windows is a strong choice for a server host due to its user-friendly control panel. Windows also features common windows dev tools within its server infrastructure making it easier to work with if you are already familiar with windows development tools. Windows is known for being one of the most stable and reliable server hosting for web applications out there and with it being used by many people around the world this leads to being easier to fix bugs, fix server issues and update the servers. Since it is not open source like Linux, it could get pricey to run your servers on windows than other operating systems. It also requires more updates and patches to the servers since it is not open source like Linux which can lead to higher costs and needing more education since Windows is constantly updating and upgrading their OS. Unlike Linux, windows is more susceptible to DDOS attacks and cybercriminals since they do not have as good of security features as Linux. Since you have to constantly update your servers and their systems are constantly changing this could lead to more resources being used and requiring more time and budget in updating and upgrading your server. Windows also does not support as many programming languages as an open-source server host such as Linux. | Mobile devices being used to host a server can be more cost effective of a way to host your server. This is because you do not need a separate device to host a server on a mobile device and needing different servers established for multiple different devices. Servers on mobile devices are more user friendly as well since they do not require any downloads on the client side and just require the client to use a web browser. Using a mobile device is better as well since it uses common coding languages and can provide the same user experience across the board for whoever is accessing the server vs using something more complex like Windows OS. Since you are using a mobile device to host a server it is also more cost effective since no new technology or software is required vs having to license a server through Windows OS or Mac. On the other side however, mobile devices require a constant strong internet connection to run a web based application which could lead to issues if someone has poor cell service or is in an area with a poor signal. If you are in an area with a poor signal this could lead to poor performance of the application since the signal will not be as strong. Since all mobile applications use the same coding language this leads to less customizability than running a full server on Windows or Linux since mobile is more uniform than something that is open source. It would also be a challenge to use a notification system for the mobile device since it is a web-based application and the user would have to be on the browser to use the application. |
| **Client Side** | Since Mac is so user friendly and well known of a OS, this leads to most users being experienced on how to use this software to open an application and run the app than something vs Linux leading to a low learning curve for new users. This can also increase user retention since most users will know how to use macOS. MacOS also integrates with appleID and other apple devices which lets users easily transfer the application data to another device running macOS leading to users constantly coming back to the application and increasing user retention. This also helps users be able to back up their data easier since they can store it onto multiple apple devices. macOS servers can be more costly to host and license leading to needing more of a budget and expenses going to hosting the client-side servers. Since macOS is used solely on apple devices it can be more difficult to host servers for other devices that are not running macOS leading to longer costs with supporting multiple types of clients and also a longer development time since we would have to teach programmers how to use macOS to host servers for non-apple devices. | Linus is a open source code based which means it is completely free to use. There are no costs associated since it is generally free to use. Linux is a stable server host and performs well for client-side server hosting. Since it is open source, it does not require any constant upgrades to devices being used to build the server or the server in general meaning that you can use older computers to build the server. Since it is open source there are many forums that can be used to troubleshoot the server and also features a vast use of different programming languages that can run on Linux. Since it is open source, it may have a steeper learning curve than something familiar such as Windows since majority of people have built something in Windows and not many people have used Linux. Since it is open source there are probably many different forms of Linux out there to use leading to people being familiar with one form of Linux and needing trained on how to use a different form of Linux which is where cost issues could come up with training people on how to use the specific Linux application that you are using. | Windows is more of a user-friendly interface than something like Linux. It is compatible with multiple different applications leading to easier solutions for various needs of a company. It is also easier to find solutions and experts with windows servers since it is used worldwide. Since it is Windows though it does have higher licensing fees for running a server and higher demands of running a server since windows OS is constantly updating. You also are not able to run Windows OS to build servers on older computers due to the OS constantly being updated which would require buying the most up to date computers or using a virtual environment to build the server. | Mobile devices for server management is beneficial since they all use a common coding language. Since the servers are centralized to mobile devices it leads to easier maintaining and updating of the server systems leading to lower costs to run the servers. This leads to a higher ability to scale the web-based application and increasing the amount of users at a low cost. There are some performance concerns for users though since not every user has the same uniform phone and data plan leading to issues accessing the client-side server down the road depending on if the user is in a low cellular service area. Since we are using mobile devices as well, debugging can be a little tricky for people who are not familiar with servers being built on a mobile device. The most expensive part of using mobile devices is that since the OS is not as simple to host a server as using a desktop computer, it can become expensive to host your server. Cost and performance can also increase as well with how much resources are consumed by hosting a mobile server instead of a normal desktop server. Since the server is not being hosted on a desktop or a computer, it could be easier to overload compared to hosting on an actual server. |
| **Development Tools** | Since MacOS is ran through apple specifically, they use a standard IDE on all of their platforms called Xcode. MacOS also uses C and C++ as the programming languages so you could use an IDE like Visual Studio or Sublime. | Since Linux is open sourced, there are many different languages you can use the build the server which leads to being able to use many different IDEs as well. Some of the common languages you can use are: C, C++. Python and Java. Some of the common IDEs are PyCharm, Eclipse or Visual Studio. | Since windowsOS is ran through Windows specifically you use their standard IDEs and supported languages. The programming languages used are C and C++ and Microsoft does have their own IDE that you use to build your server which is Visual Studio. | With hosting a server on a mobile device, you are presented with two different operating systems: Apple OS and Android OS. If you choose to build the server on Android OS, you would use Java or C++ while if you use Apple OS, you would use C++ or C. The IDEs that support these are Android Studio, Xcode, Visual Studio, Eclipse and IntelliJ. You also have some mobile app development tools you can use to build your mobile app such as React which is a framework to help you build your application to be used on multiple different platforms. |

## 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 using WindowsOS which has advantages of being able to be accessed by a large population of gamers while also being extremely well protected and secure. Windows operating systems is used by majority of game developers and is known to have excellent support for a wide range of devices across many different areas of gaming. Just about every computer runs on Windows meaning that it will be easy to edit code and run updates on the server as long as everyone is using either Windows’s or a Windows Virtual Environment to make updates to the server.
2. **Operating Systems Architectures**: Windows has a broad range of software it is compatible with making user access easy and editing to the game just as easy due to the large support volume that WindowsOS has. Windows 11 also features an in-house gaming performance mode where your background activities are turned off or slowed down (notifications, Spotify etc.) to prioritize more RAM going toward the game you are playing resulting in a better performance overall during your gameplay. Windows also supports enhanced audio features making your gaming experience more immersive. WindowsOS also comes with Visual Studio which is Microsoft’s in house code editing tool that runs in C++ which is an advanced performance language. Due to Microsoft owning both WindowsOS and Xbox Studios, integration towards a console release on Xbox is fairly simple to do since it can all be done natively in Visual Studio leading to less development time and more simplification of the process of exporting games to other platforms.
3. **Storage Management**: When running a game through Windows it is highly recommended to install the operating system that the game is running on onto an SSD while storing physical games on your HDD because the SSD is a solid-state drive which has better performance, load times, smoother graphics compared to running the same exact item on an HDD. Windows also has a feature called Storage Sense which will auto free up storage by deleting temporary files, auto uploading items to the cloud and empty your storage bin for you automatically.
4. **Memory Management**: Windows has a feature that tracks your overall RAM usage and helps you decide which items get the RAM allocation. It will automatically move RAM to things that are running based off of a who needs more RAM situation and then as things stop running it will move the RAM to things that are still running. Windows also has the ability to use Virtual RAM when your physical RAM is in full use which helps you run larger programs easier and does not consume as much time. Windows also has a performance monitor which helps you see in real time how your memory is being allocated.
5. **Distributed Systems and Networks**: Windows does have the ability to host servers on multiple platforms for gaming (Xbox, PS5, Switch and PC) and it does this by setting up network protocols to ensure that every version of the game gets the same experience. You can create a dedicated server for every platform to connect to when playing the game and then from there you would have everyone log into the game with a Microsoft account so this way regardless of how the game is being played, as long as you have a Microsoft account you are able to play the game. You could also set up peer to peer connections where someone who connects to the game is automatically the “host server” and people connect to that specific person which would require a dedicated system built out in the server for this process.
6. **Security**: To protect user information such as log in information one could encrypt that information deep into the server via a kernel level drive. You could also set up Authentication systems and permission systems where only people who are eligible to access certain information have access to it so this way someone who works in customer service to handle refunds does not have access to the source code of the game. This protects from information being leaked by upset employees by only giving that information to the most trusting of people. You can also enable firewalls to protect from hackers and other data breaches that could happen. You could also set up Network Segmentation which would make it so each server is separate from one another so if one server goes down (let us say the cross platform connections) then it would not affect the source code at all which would in turn reduce the threats that can come from network based attacks.