

<Name-of-Software-Application>

# **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 | 01/23/24 | Braden Whitcher | Wrote explanations for executive summary, design constraints, and domain model |

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)

Our client, The Gaming Room, wants to develop a web-based game based on their current Android app, Draw It or Lose It. The game works by having an image being rendered to the screen within a 30-second time span, and the team must guess what the image is before time expires. If the current team does not guess the image, then the opposing teams are allowed one guess with a 15-second time limit. The game must be able to serve multiple platforms. One way we could solve this is by making the game responsive on all devices, phones, tablets, laptops, and desktops. This way, anyone can play the game no matter the device they are using.

## 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)

One major design constraint is designing the game, so it looks good and provides a good user experience on all devices. This can be tricky because if you put too much stuff, or make the design too complicated, on a desktop, then it is going to be hard to implement that for phones and tablets, because of the reduced screen size. Another one is since an image is being rendered in 30 seconds, the designers and developers must make sure that the load time for that image is accurate and make sure there isn’t any latency as that could have a major impact on the experience of the game. One final design constraint could be to make sure the code the developers write is compatible across multiple web browsers. All web browsers have their own way of deciphering code, so making sure the website works well among most web browsers could be crucial.

## [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 the UML diagram, there are 7 classes. The classes are ProgramDriver, SingletonTester, Entity, GameService, Game, Team, and Player. Please note that none of these are objects, they are classes in which an object is made from them. The “Entity” class is a parent class and “Game”, “Team”, and “Player” are child classes to the “Entity” class. “GameService” is not a child class, but it is associated with the “Game” class, which is denoted by the “-----0...\*-----” line. You will notice that with the “Game”, “Team”, and “Player” classes as well, they are all associated with each other, and all have the same association line. The “0...\*” is the sign for multiplicity, which means that the class can have 0 or many instances. Each class has three sections, one for the title, one for the attributes, and one for the functions. Attributes represent the information or data for that class, and functions represent the functionality that can happen in that class; or other classes. For example, the “addPlayer()” function in the “Team” class adds a player from the “Player” class to the team.

**"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 | One advantage of Mac is it is a Unix-based operating system. Unix-based operating systems are known to be more secure and are less susceptible to viruses. Another benefit of it being Unix-based is that it offers a powerful command line interface, which is helpful to developers. A weakness is that it could have some compatibility issues compared to Windows and Linux with certain technologies. | One characteristic of Linux that makes it stand out more is it is open source. This brings a lot of attention to developers or people who want to contribute to big projects to help make them better. This can also turn into a major advantage to businesses, because since it is free and open source it reduces licensing costs. Another good advantage is its powerful command line. You can manipulate things such as permissions, servers, and interacting with databases. One disadvantage is if you are used to using Windows or Mac, it could be hard transitioning and getting used to Linux. I have never used Linux so I wouldn’t know, but this is what I have heard. | Windows is a very popular operating system and is used by many people. A good characteristic is it has a good GUI, and an easy file management system that many people enjoy. A good advantage is if the web-application is built using Microsoft technology, then hosting on Windows server can enhance performance and compatibility. A disadvantage is, unlike Linux, licensing can get expensive, which can be a problem for the budget. Another one is Windows can be susceptible to more security issues. There have been a lot of problems in the past with viruses, but it has gotten better. | Mobile devices, compared to laptops and desktops, are unique in their capabilities. One characteristic and advantage of mobile devices is they can access any web-based application from anywhere and at any time. This means that it can help drive more traffic to the software application. A weakness of them is the vastly smaller screen size compared to tablets, laptops, and desktops. This can greatly impact the design and would need to be communicated with the developer, so they come to a conclusion. Another weakness is the smaller amount of processing power and storage. If there is too much stuff going on in the application, then this can greatly affect the user experience. |
| Client Side | One major thing to consider is making sure the web-based application is cross platform compatible. This doesn’t only apply to Mac, but to all operating systems. Doing this will ensure everyone will have a good user experience. I know that Mac does have some specific guidelines to follow that are different from other operating systems, which is something the developer should keep in mind. | A lot of people who use Linux are used to using the command line a lot. Implementing a feature where they can interact with the application via the command line wouldn’t be a bad idea. You have to keep in mind the different Linux distributions, so you may need to develop some features to where it can adhere to that. One example is the different kinds of file systems, some have different permissions and types. | One big consideration is making sure the web application is responsive on all Windows devices, since there are desktops, laptops, and tablets that all run on Windows. In addition to that, the version of the Windows software needs to be considered as it could have an impact on the experience of the application. | One software consideration is the need to program it in multiple languages to deliver the web application in an app. Responsive design also needs to be considered because of the different screen sizes, and for portrait and landscape mode. It would not be good if the application looked bad in either mode. Good security features should be highly thought of to adhere to each mobile platform, so they are adequately protected. |
| Development Tools | Mac comes with an IDE built into it called Xcode, which can be beneficial for server-side development if the developer is working on mac specific applications. Also, if an IOS app is made from the game, the developer would either program it in Swift or React Native. Swift was developed by Apple and is what they use for IOS, macOS, and watchOS apps. | There are numerous programming languages that can be used on Linux, such as C, C++, Python, Java, Rust, etc. The same thing applies to IDEs, such as Visual Studio, Visual Studio Code, Eclipse, etc. Other tools, besides IDEs, can be used as well, like Git and Docker. | Just like Linux, Windows has a big variety of programming languages and tools to use. A few of them are C#, Python, C++, Java, Visual Studio, Visual Studio Code, Eclipse, and .NET CLI. Many of these programming languages and tools were developed by Microsoft. | There are many programming languages that are specific to certain mobile operating systems, and some that are cross compatible. Swift would be used to develop the application for IOS. Java/Kotlin would be used to develop it for Android. React Native would be used to deploy the application on both IOS and Android. |

## 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.>
2. The operating system I will be choosing for The Gaming Rooms web application, Draw It or Lose It, is Linux. One main characteristic of Linux is that it is open source. It has a powerful codebase and an active development community to help improve and fix any bugs. It is highly scalable which means it can handle large amounts of traffic and data. It is compatible with multiple hardware and software technologies such as x86, SPARC, and ARM. Linux servers are high-powered and built to withstand the most challenging business applications, which is important, especially for web services.
3. Operating Systems Architectures: <Describe the details of the chosen operating platform architectures.>
4. Linux is a Unix-based operating system. Unix was developed in the 60’s by Bell Labs and is a multi-user, multi-tasking operating system. The main component of Linux is the kernel. It was developed to provide a free, or low-cost, operating system to individual users. A very unique characteristic of Linux is the use of different distributions. A distribution is an operating system package built around the Linux kernel. The kernel is the one that interacts with the hardware, and the distribution creates a user-friendly environment. Some common distributions are Linux Mint, Ubutu, Solus, and Manjaro. The main components of Linux are the hardware, kernel, shell, applications, and utilities. The shell is used to interact with the kernel. Users would enter a command, which is then performed by the kernel’s function. The shell and command line are very powerful tools and are what many people like about Linux.
5. Storage Management: <Identify an appropriate storage management system to be used with the recommended operating platform.>
6. For this application, I think using cloud storage is one of the better options because it is an effective solution for storage management and offers good benefits. Anyone can access their data over the cloud no matter what device they are using or where they are, if they have an internet connection. The cloud provider you use backs up the data for you, so there is no need to use any external devices. Another benefit is that it can cost you less than having an on-site server infrastructure. The business does not need to pay a professional since it now relies on the cloud service provider for any technical support. Another important aspect is that you only pay for the storage you need. If you need more or less storage, then it can be easily increased or reduced. One final benefit is that cloud service providers offer good data security, so your users are adequately protected. Any data sent over the internet is encrypted and the storage is password protected.
7. Memory Management: <Explain how the recommended operating platform uses memory management techniques for the Draw It or Lose It software.>
8. Proper memory management is crucial for the game, Draw It or Lose It, because images can be retrieved faster from a computer’s memory than from storage. Linux uses a certain memory system called virtual memory. Virtual memory makes the system appear to have more memory than there is by dividing it with other competing processes, a process is a program that is actively running in an operating system. This system offers good protection by having each process in the system allocated to its own virtual address space. Each space is separate from one another so they cannot be affected by other processes running other applications. It also has a powerful tool implemented called memory mapping. Memory mapping is used to map certain files and images into an address space. This is good for this application because it is fast and efficient and is needed for the images to be fetched and rendered continuously over a 30-second time limit.
9. 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).>
10. Distributed systems are a collection of physically separated computers connected to one centralized network. The centralized network is the one equipped with the distributed system software. The distributed system software allows the computers to organize their activities and share their system resources such as data, hardware, and software. Each computer will communicate with each of the distributed systems, and they will share any files and resources so they can complete their tasks. A database is used to store any data processed by a distributed system connected to the central network.
11. 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.>
12. Linux is a great choice for any of your security needs. Since it has a very large open-source community, people regularly test and fix any security vulnerabilities they find. Companies like Apple and Windows keep their code hidden, which has some flaws because when there is a security issue, only a small team of developers is working to fix it rather than a large community of people. Security features are built into the system itself, making it one of the most secure operating systems. Some of these features are firewalls with packet filters, AppArmor mandatory access control, and UEFI secure boot firmware verification mechanisms. Linux also has a strict user privilege model and offers a wide variety of built-in security defenses for the kernel to protect against cyber-attacks. The user privilege model is a system that gives user accounts certain permissions to control access to specific actions on a computer system in a multi-user environment. This gives a person just enough permission to complete their tasks. It also makes it more difficult for software to spread malware. Besides built-in security features, a good way to protect user information across various operating systems is by requiring the user to create a long password with special characters. For example, a password between 8-20 characters long with at least one uppercase, lowercase, and a special character. This greatly reduces the chance of software to figure out a user’s password.

Resources:

*9 reasons Linux is a popular choice for servers.* (2023, February 28). Logic Monitor. Retrieved from <https://www.logicmonitor.com/blog/9-reasons-linux-is-a-popular-choice-for-servers>

*Architecture of linux operating system.* (2023, September 25). Geeks for Geeks. Retrieved from [https://www.geeksforgeeks.org/architecture-of-](https://www.geeksforgeeks.org/architecture-of-linux-operating-system/)linux-operating-system/

Secure Storage Services. (n.d.). *Pros and Cons of Cloud Storage.* <https://www.securestorageservices.co.uk/article/11/pros-and-cons-of-cloud-storage>

TLDP. (n.d.). *Chapter 3 Memory Management.* [https://tldp.org/LDP/tlk/mm/memory.](https://tldp.org/LDP/tlk/mm/memory.html)html

Day, B. (2023, May 19). *How Secure Is Linux?.* Linux Security. Retrieved from <https://linuxsecurity.com/features/how-secure-is-linux>

*What is a Distributed System?.* (2024, February 11). Geeks for Geeks. Retrieved from [https://www.geeksforgeeks.org/](https://www.geeksforgeeks.org/what-is-a-distributed-system/)what-is-a-distributed-system/