

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template**](#_l6ti7uoag22u)1

[**Table of Contents**](#_30j0zll)2

[**Document Revision History**](#_grjogdjh5fi8)2

[**Executive Summary**](#_sbfa50wo7nsh)3

[**Design Constraints**](#_2et92p0)3

[**System Architecture View**](#_ilbxbyevv6b6)3

[**Domain Model**](#_8h2ehzxfam4o)3

[**Evaluation**](#_2o15spng8stw)3

[**Recommendations**](#_m8aleynsvzvc)5

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 07/18/21 | Fabian Rodriguez | Initial Revision |

**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 software design should be based of a four round guessing game where different teams must guess what is being “drawn”. The client was to extend the functionality withing the game. This functionality includes multiple team’s gameplay while the player has unique ids. Only one instance of the game may be allowed at a time, and identifiers must be assigned to each game, team, and player.

## [Design Constraints](#_2et92p0)

The major design constraint is to create a proper workflow that allows the user to access the data from any platform without any roadblocks.

## [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 below portraits the classes related to “The Gaming Room”. The diagram displays the following OOP principles: Game, Team and Player inherits from the Entity class; therefore, those classes share the attributes. This approach helps us to keep our code dry. We can also see encapsulation as “Entity” limits access to public methods in the program.

****

## [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** | Although Mac is expensive it is a great solution for hosting web-based software. It secure and rather simple to setup. | Linux is one of the most used to OS to host web-based software. It’s relatively economic and is also simple to deploy software. | Similar to Mac, windows is rather expensive. Windows has a plethora of software to make dev life easier. | Mobile devices are not as powerful as different operating systems to deploy web-based software. |
| **Client Side** | Amount of time for Mac would be similar or lower than Windows. It is an expensive OS. An average knowledge will suffice to use this OS. | Linux has a higher learning curve. Cost would be the lower compared to other OS. An expert on Linux must be required. | Windows is probably the easiest one. The cost might be more expensive than Linux system, however, is lower than Mac OS. | Mobile device support should be one of the most complicated due to the diversity of OS. An expert on mobile is required to perform task efficiently. |
| **Development Tools** | Relevant Programming languages are C#, Swift, Python, JavaScript and more. There are a lot of IDEs/ tools for developers that are compatible for different platforms such as VS Code, Jetbrains IDEs(Intellij, Pycharm, CLiom) | Relevant Programming languages are C#, Swift, Python, JavaScript and more. There are a lot of IDEs/ tools for developers that are compatible for different platforms such as VS Code, Jetbrains IDEs(Intellij, Pycharm, CLiom) | Relevant Programming languages are C#, Swift, Python, JavaScript and more. There are a lot of IDEs/ tools for developers that are compatible for different platforms such as VS Code, Jetbrains IDEs(Intellij, Pycharm, CLiom) | Relevant Programming languages are Kotlin, Swift, Dart, JavaScript and more. There are a lot of IDEs/ tools for developers that are compatible for different platforms such as VS Code, Jetbrains IDEs(Intellij, Pycharm, CLiom). A few frameworks that can be used are Flutter or React Native. |

## 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**: Linux would be the suggested operating platform. Although Linux can be complicated, there are a lot of resources that can be used to facilitate the development of Draw It or Lose It.
2. **Operating Systems Architectures**: Linux boast a huge supportive community which is very helpful when trying to learn new things or work around others. Being able to use terminal for configuration is a plus.
3. **Storage Management**: Linux is very helpful when it comes to storage management. Also there are a lot of Linux based cloud storage services that can be leveraged for our purposes.
4. **Memory Management**: Linux includes implementation of virtual memory and demand paging, memory allocation both for kernel internal structures and user space programs, mapping of files into processes address space
5. **Distributed Systems and Networks**: The best way to manage a system than connects to other different platforms would be to use an API that can be accessed from those platforms.
6. **Security**: Like most system, Linux can help our user data to be safe and secure. To add more security, we can encrypt the data request with TLS, use strong authentication and authorizations. Practice the principle of least privilege, don’t expose data that is not necessary and remove information that is not meant to be shared.