

# 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 | <06/19/2021> | Michelle Hitchcock | To ensure this game is compatible with the hardware this upgrade is focused on the software of the game |

**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 design problems presented by Draw it or Lose It include ensuring that the developer works within the limits of the software and the designated design template laid out. Keeping in mind the idea of later expansions such as hardware and ensuring all software requirements laid out by the client are met.

## [Design Constraints](#_2et92p0)

Some of the design constraints that are specific to this project include working with the singleton method and working primarily alone. More web-based constraints might include things like having to consider the main operating system that the application is designed for, as well as who the audience is. These constraints can have an impact on what features are included and even what language is used to create the application.

## [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 below we can see that all the classes are a part of the com.gamingroom package. Furthermore, we can see that Game, Team, and Player are inheriting properties from the class Entity. There is also an association between GameService and Game. Something else that can be seen between GameService, Game, Team and Player is the 0…\* which indicates that these classes have 0 to many instances. Finally, ProgramDriver and SingeltonTester have a composition relationship. When looking at this diagram the object-oriented principle of abstraction is clearly seen.

****

## [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** | It is expensive, however, it has the benefit of more security and the ability to work well with the apple ecosystem. | It is in general more difficult to use than windows and mac, more of a niche audience. | Windows has ease of access on its side and is readily available. | Highly popular, however there are also many development  constraints |
| **Client Side** | The cost of the Mac means that the market is often smaller for Mac programs. Programing costs might also be higher, however there may be some benefits as clients might more interested. | Since Linux is a more niche market the programing and expertise costs might be higher, however, interest might also be higher. Development on this OS is generally more open source. | Developing on windows would likely be one of the cheaper options, with the downside to having it also be easily overlooked. Clients might have little interest in another windows program | There is generally high demand for mobile applications, however, properly developing one can be difficult. Clients might be most interested in this option. |
| **Development Tools** | Swift is used to developing on Apple, this is also true for IOS. The apple developer tools would be the best option although there are other IDEs available. | Python and C++ are highly mentioned. Because Linux is opensource there isn’t one true answer in this case. IDEs include pycharm and eclipse. | Windows applications are written in C#  IDEs of note would include eclipse and Microsoft visual studios | When working with Apple IOS Swift is used. The Apple developer tool is an IDE as for other phones such as Samsung and android it appears to be a combo of C, C++ Kotlin, Python and such, as there are many different mobile platforms it is hard to give a complete answer to this. |

* 1. **Does each of the operating platforms offer a server-based deployment method where the website will be hosted?**

Each operating platform offers server-based development methods, or in the case of mobile the mobile equivalent, where the website can be hosted and the game can be run.

* 1. **What are the potential licensing costs to the client, The Gaming Room, for the server operating system?**

When looking at servers there are some options. The company could choose between maintaining their own servers or opting for paying for webhosting. To buy their own, the price would likely be upwards of $3,000+ for one server. whereas webhosting can range from $3-$300 per month. So, the traffic and complexity of the website would need to be taken into account. With all of these costs to the company taken into account this creates a scenario where the client might be asked to pay a fee to play, pay to purchase add-ons and features, or pay for more playtime. The final amount of cost to the company would ultimately determine these prices.

* 1. **What is required of the application development process to ensure the application is compatible with all web browser platforms and mobile devices?**

It is important that the application development team does adequate testing to ensure that the final website runs on all needed browsers. Ensuring that the application runs and is compatible (e.g. fits in the window and has the features accessible) is important and is best accomplished by systematically testing.

* 1. **What impact do these technical requirements have on a development team? Consider whether multiple development teams may be needed.**

This is a cross platform project that will require systematic testing to ensure that all parts are working in unison. I suggest having a single, knowledgeable team working with an agile approach to get this done in a timely manner. This is because with the whole team working together, but on their own parts, any issues can be spotted. Something that is important for a complex project like this.

* 1. **Are there licensing costs related to the development tools?**

There are licensing costs for some of the development tools. For example, to publish an application on any apple products (IOS, MacOS etc.) enrollment in the Apple Developer program is required. This is a $99 annual fee. Similarly, there is a fee for Windows as well which is $99 for a company, however this is a onetime fee rather than an annual fee, because of this it is ultimately less expensive. Linux is open source this means that developing on it would not cost an extra fee making it the cheapest option.

## 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**: In order to have the option to expand Draw It or Lose It to other computing environments, my recommendation is Windows. By developing the app for Windows the development team would be able to port the app to Linux. For example, C# is a common language used to develop for Windows and because Linux apps utilize many languages including C++ (a sister langue to C#), this leaves the option to expand to this system open. Likewise, because the mobile app field has many languages available, developing the app for the windows platform leaves the opportunity to expand to those platforms as well. This crossover between all three section leads, plus windows ease of entry, leads me to suggest Windows as the best operating platform.
2. **Operating Systems Architectures**: Windows operating system is working with an unprotected kernel. This has both benefits and drawbacks. This may make some parts of network development easier, however, it can pose difficulties when dealing with security risks. Working in this environment can make it easier to develop programs and software for this system. Because of this relative ease, security flaws are often easier to find and exploit as well.
3. **Storage Management**: An appropriate storage management system for windows would be physical backups such as hard drives/severs alongside some potential cloud storage for ease of access. Given that there will be many different assets and files ensuring all data is adequately backed up is important.
4. **Memory Management**: Although this application is starting out small, because of the proposed growth for the Draw It or Lose It game I suggest the “Paging” memory management method. This is because as the game grows and the ability to have more players, more teams, and more information allotted to memory this method will allow for physical address to be stored noncontiguously, negating some of the fragmentation issues with other methods. This memory management method will allow for growth over time. By utilizing this method user experience may be improved due to improved loading speeds.
5. **Distributed Systems and Networks**: Because of the potential growth of the network associated with this game, my suggestion is building a strong connected network that has built-in contingence plans for network outages. For example, ensuring that if/when an outage does occur the correct people are notified, users are given a notification message (site outage page) and all data is protected.
6. **Security**: Windows is not the most secure of the choices for operating systems, this will necessitate more security being incorporated in the development of the network and the application itself. It is important that the end user has security as well as the company.