

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 3.0

## 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 |
| --- | --- | --- | --- |
| 3.0 | 02/20/21 | Chris Harvestine | This is the third draft of the design template. This draft includes a summary, design constraints, development requirements and more. |

**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 name of the game we are making is called Draw It or Lose It. In this game team members try to guess what is being drawn. Instead of drawing images on an easel to help team members guess the object, the game will render the images from a library of drawings as clues. Drawings are rendered at a consistent rate and are fully realized at the 30 second mark. If the team does not guess correctly before the time expires, the remaining teams will get a onetime 15 second chance to answer.

## [Design Constraints](#_2et92p0)

Currently the game Draw It or Lose it is only available on android. Our client The Gaming Room, wants to develop an online multiplayer web-based game that is compatible with the major platforms. Instead of programming for each operating system the idea is to make the game web based.

## [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)

By looking at the UML we can understand many things about the game mechanics. A game will be able to have one or more teams, each team will have multiple players assigned, game and team names will have a unique identifier to avoid overlap. As for the logic of the game, the computer will render an image while one player tries to draw that image. If the team member guesses correctly within 30 seconds, they receive a point.

****

## [Evaluation](#_2o15spng8stw)

Each operating system has its strengths and weaknesses, and I will cover some of them here.

**Strengths:**

1. Windows: Windows has been around since 1985 and many of the new PC’s that come out have this operating system on it, thus Windows has a large user base. The Windows operating system also easy to learn and use while offering great support for new hardware.
2. macOS: macOS is the second most used operating system and has fewer virus and malware attacks than Windows. The macOS operating system has a similar GUI for their programs and can also run Windows.
3. Linux: Linux has great security since each application needs to be authorized by the admin user. The operating system is very lightweight and does not require high powered hardware to run. It is free to use, anyone can download it. Since Linux is open source, there are a wide variety of programs to choose from, some are updated regularly.
4. Mobile Devices: Ease of use, these operating systems are intuitive in design due to touch controls, almost anyone can learn to use them quickly. Mobile devices are very popular which means a high user base. Mobile devices are increasing in ram and processing potential with each new iteration. Although mobile devices have not caught up with PC’s the quality of games that can be played off mobile devices is increasing.

**Weaknesses:**

1. Windows: A high resource requirement, you need a decent computer to run the latest version of Windows. The Windows operating system can also use a decent amount of hard drive storage when updates are applied. Poor security, Windows computers are more vulnerable to attacks from hackers.
2. macOS: Limited and restricted availability of applications can make this operating system less desirable. Limited game options combined with hardware upgrade limitations makes this platform less desirable for gamers.
3. Linux: Because the software is not as popular as Windows, it can be hard to find software to support your needs. Not all companies write drivers for Linux therefore you may not be able to use all your hardware with this operating system.
4. Mobile Devices: Although mobile devices have increased in hardware capability, they are still far behind dedicated gaming computers. If you are creating a graphically intense game, you need to look at the current capabilities and possibly scale back your game to meet the hardware limitations. Mobile device internet caps may also become a problem if you are creating a multiplayer game. It will be important to optimize your game data transfer to work with in these limits.

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Malleable server commands to construct the server which helps with access and making changes.  Mac Server can be expensive to start up compared to Linux and Windows. If using Mac clients on the network, the features are easy to implement, comparable to Win group for Mac clients. | Linux has enhanced security features on the operating system as well as the server. Linux is open source keeping maintenance and costs low. Considered to be stable secure and flexible. However, the update process can be complex and not all professional programs work with Linux. | Beginner friendly, intuitive operations through GUI. Supports a large number of third-party applications. Guaranteed long term support. Negatives include security related errors; high licensing cost and high resources might be needed because of required GIU’s. | Developing a client server application can be challenging because of the mobile device’s limitations. These limitations can include network traffic and battery life consumption. A mobile web server application allows hosting of web applications, pages and server-side control. The mobile web server allows optimization and synchronization across many platforms regardless of type. |
| **Client Side** | This platform will require an average amount of time required for development, an average amount of expertise required, and the cost can range from medium to high. Development on this platform will be even across the board aside from expertise required. Since the cost can get into the high range, we can offset this by taking the time to learn the skill required instead of paying large amounts of money to a programmer with this skillset. | Highest amount of expertise required; amount of time investment needed is Higher as well. Cost is the lowest of these solutions. Time and cost may go up because of higher expertise required. We will need to recruit skilled programmers of Linux. | Windows requires the least amount of time needed to invest. Windows also requires the lowest level of expertise. The cost can range from medium to high. This might be the ideal situation where we can save time on required knowledge and time required by paying a little more for the infostructure that is in place. | Costs can vary given how much data is calculated on the phone and how much is calculated on the server. I see the costs around the medium range. A higher level of expertise is required to implement this kind of option. Given internet data caps for phones I’m not sure this is a good option for users. This is another high skill gap requirement weather we take extra time to learn the environment, or we pay extra money for programmers with that skill set. |
| **Development Tools** | Swift will be the default environment to use when developing this game. I suggest using Atom as the integrated development environment with in swift.  Objective -C  Docker  Gitlab  PyCharm  Eclipse  Visual Studio  Html/CSS  PHP/SQL  Swift | Linux offers a wide variety of support for most programming languages.  C, C++  CSS  Java, JavaScript  Perl  Python  Docker  Gitlab  PyCharm  Eclipse  Visual Studio  Brackets  Html/CSS  PHP/SQL | Microsoft heavily uses C++ in most of its core applications. However, there is also a wide variety of support in other languages featured below.  Docker  Gitlab  PyCharm  Eclipse  Visual Studio  Notepad ++  Html/CSS  PHP/SQL | Swiftic for IOS has been labeled as the everyman application maker thanks to its easy to navigate interface. It is currently one of the most popular platforms that has led to the creation of over one million apps.  For android development, Java is the official programming language. Android was built off Java so there are mana Java libraries at your disposal. You may also use the following languages featured below.  Docker  Gitlab  Dcoder  Eclipse  Visual Studio  Python  Html/CSS  PHP/SQL |

## Recommendations

1. **Operating Platform**: PC’s still have the greatest hardware capability compared to mobile devices. The development on Windows is beginner friendly on the client side and requires the least amount of time invested on the server side, this is a big plus for development. Windows provides many tools to choose from, which means a preferred programming language is supported, this helps the programmer start the development process quickly. The windows 10 user base is quite large as over one billion individual users have logged into Windows 10. This large user base will help the game receive large amount of exposure. Windows supports many different types of hardware ensuring an increasing number of users for the future. For these reasons I would choose Windows and the PC for development.
2. **Operating Systems Architectures**: ARM processors for mobile devices, x86 architecture for laptops and PC’s.
3. **Storage Management**: Since I am choosing a standard PC computer, I would go with the 7200-rpm hard drive. 7200-rpm hard drives still come standard on most current PC builds outside of dedicated gaming PC’s which tend to come with SSD’s. The size of a hard drive in a computer purchased recently is around 500 gigabytes, this is more than enough space for this game as I believe the game to come in around 2 to 3 gigabytes currently. This game should offer support for SSD’s as well, they have become more cost effective and more users will have them. SSD’s will help the game load the assets quicker into memory which reduces game load times.
4. **Memory Management**: Windows memory management system implements a demand paged virtual subsystem, which is another way of saying it is a lazy allocator. If you open a program in Windows, it doesn’t open the whole application and DLL’s into physical memory, it does so as the application needs it. This increases efficiency of the memory usage and allows less powerful hardware play current games and applications. Windows in 32-bit mode can address up to 4 gigabytes of memory which is the lowest amount of memory needed to run the current PC operating system Windows 10, most users should at least have this amount of RAM if they are operating on Windows 10.
5. **Distributed Systems and Networks**: Windows Operating system offers distributed systems that ensure consistent communication between servers and between multiple workspaces. Distributed systems and networks will allow this company to perform high level computing with off the shelf computers with microprocessors, this decreases the upfront cost for the developer. Using distributed systems and networks will increase our programs flexibility and cost effectiveness this this and future titles. There are some disadvantages to these systems namely they can be vulnerable to hacking a sabotage due to the transfer of information on a public network. To combat this, we will use VPN’s for communications between networks and finally to the end user. Using VPN’s to communicate with the end user will also add an extra level of security for the user particularly with sensitive information.
6. **Security**: Security is very important to any program or application. There needs to be a level of trust for any client to use our system. Personal information as well as financial information may be used in this program therefore, we must protect the user’s data appropriately. Encryption is a valuable tool when protecting a user’s data on a local drive. Android, Windows and IOS devices all support local encryption, the development team will take the time to look up the necessary steps per platform for use of storing important information. A simple way the servers can transmit secure data to the users will be through a virtual private network or VPN. I suggest licensing a commercial VPN that supports all the platforms the client wants to develop for. IPVanish is very highly rated at 9.4 out of 10 and can cost as little as $2.62 per month for 12 months. IPVanish supports PC, Apple, Linux, and Android. This VPN supports all the platforms suggested for this game in addition to others, this gives us an opportunity for development on platforms not considered.