

Draw it or Lose it

# **CS 230 Project Software Design Template**

Version 1.4 (Module 8)

Final Portfolio Version

## 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 |
| --- | --- | --- | --- |
| 1.1 | 09/17/20 | Brad Byard | Software updates to finish project 1 |
| 1.2 | 10/02/20 | Brad Byard | Operating Platform and evaluation added |
| 1.3 | 10/12/20 | Brad Byard | Developer Recommendations added |
| 1.4 | 10/21/20 | Brad Byard | Final Portfolio Version |

## [Executive Summary](#_sbfa50wo7nsh)

Developing a web-based game that serves multiple platforms called Draw it or Lose it, which is currently available as an android app only.

## [Design Constraints](#_2et92p0)

* Serves multiple Platforms
* Render images from a large library of stock drawings as clues
* Four rounds of play, one minute each
* Drawings rendered at a steady rate and fully complete after 30 seconds
* If puzzle isn’t guessed before time expires, remaining teams will have an opportunity to offer one guess each to solve the puzzle with a 15 second time limit
* Allow for 1+ teams with each team allowing for multiple players
* Unique Game and Team names for each team allowing users to check if name is taken
* Only one instance of the game can exist in memory at any given time

## [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 Entity class is a base class introduced to hold common attributes and behaviors. Game, Player, and team Classes will inherit or “extend” the Entity class, and allow for only one instance. This is shown below in the UML diagram provided. The singleton class provides that only one instance exists. The code allows for the check to see if anything with current name or ID “Entity” exist already. The program driver is run to implement all these.

****

## [Evaluation](#_2o15spng8stw)

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Mac devices seem most secure but as far as use of web-based applications but I would not recommend Mac. MAC does have some free built in server applications but not sure if they would be ideal for this project as they aren’t made for high traffic. | Linux is more open source and customizable then windows and Mac. Market share seems to prefer Linux over windows as a web based game host. Good for business use. Linux uses pipes as a simpler way for processes to communicate with one another | Windows comes on most desktops so the popularity there is nice. Windows is pretty secure, but with more users comes more hackers and malware. Good for web based personal use, customizable use, gaming, business use, and high traffic use also. | Keep in mind that mobile architecture generally only works on mobile. But mobile apps are a growing and explosive market in the gaming world that more people probably own than a PC or a console right now. |
| **Client Side** | Remember that Mac is often the most expensive OS out of these three. Mac runs secure but you pay the price for it. To host you may have to register with Apple as a developer and pay thousands for software and compatibility and testing software. | Linux is open source and free so the project is cheapest with this approach. I would recommend it for cost but also, might take more expertise and knowledge to fully configure, but the possibilities are endless for a game host setup. | Window is pretty affordable and has many programs that allow server client setup for a game server, but it will cost more than Linux. Many cheap cross browser testing applications that determine product works on all browsers and mobile devices | Mobile client and server architecture is common, and would work well with this application. A mobile release may be costly because I would recommend a release on android and IOS for any mobile app. Game would be used on app not on web browser preferably |
| **Development Tools** | Most programming languages and IDE’s can be implemented on MAC but are more readily available and usable for windows. | Most IDEs and programming languages can be implemented on Linux with the proper expertise, as Linux is open source. | Microsoft visual studio is growing as a top IDE for programming and is readily available on windows. | Keep in mind that Java is the main language for developing android apps. A java program that was released on windows, could be modified to work here. |

**Developer Recommendations:**

I recommend a digital download release (via the web) for a windows server-client application release with an Android-IOS mobile application release also. With these platforms I think you will have the most success. Some data will be accessed online so a web connection will be needed for the game.

1. **Operating Platform**: To get maximum use out of your product as a web-based program I would recommend releasing it on Windows type server. It will be generally affordable, not the cheapest option but I think the most usable. I think it would do well as an app available on IOS and android also. Mobile phones are devices that people use every day, more so than their computer. And when they surf the web or play a game, it’s much easier to click an app than to navigate to a website. But as a web-based game where you go to a website, I would recommend windows or also for a digital download.
2. **Operating Systems Architectures**: The architecture for windows should be fairly common. The client/server setup that will be used here is very common already in many games. Below is a layout from Microsoft on windows architecture.

A picture containing graphical user interface

Description automatically generated

Source: (sulung, 2018)

1. **Storage Management**: Windows storage management is optimal and works great. The physical storage of this game will need to be downloaded onto physical storage space on a system. This could be a disk drive, or a Solid-State hard drive. The game could be smaller if we put the game on a disk and ran it from it, but in today’s world I would just recommend a digital download for the game, as that seems to be the most used and most common approach. Regardless of the operating system the game is released or coded for, I recommend releasing the game on a game engine like *Steam,* where people can go download the game onto their systems physical memory. On mobile devices this game would be downloaded from Apple’s App Store, or Google Play on Android. On mobile devices games tend to be the apps that require the most storage. These places listed are places where most people go to download the game to storage on their device. We have to keep in mind this game needs the space to host servers and big lobbies of online players. Since the game will be focused with an online connection needed (multiple players connected in lobby) and images will roughly occupy 1.6GBs I would recommend storing those images online, and having the game access them via network connection. Let’s consider a similar mobile game such as the current sensation game Among Us. This game hosts online lobbies similar to that of Draw it Or Lose it. This game occupies 156 MBs on iOS so I would shoot for having our game be similar in size and we can reduce the size by storing images online and accessing them during gameplay.
2. **Memory Management**: Memory compression is a Windows memory management technique that was introduced in Windows 10 and it was designed to improve responsiveness of the operating system during time with extreme use. 64-bit Windows has a virtual address space of 8 terabytes and all threads of a process can access its virtual address space (The GIS Encyclopedia, 2018). Windows provides up to 16 TB of addressable memory space divided equally between the kernel and the user applications. With 16 TB of physical memory available, 8 TB virtual address space will be allocated to the kernel and 8 TB VA space to user application memory (Microsoft Corp, 2018). Windows memory management and use of virtual memory will make this game run at high speeds and be able to handle it well.
3. **Distributed Systems and Networks**: If we distribute this game coded in the Java language it will be easy to make it work between various platforms. This is due to the fact that people writing game code usually use Java, and java IDE’s are already available on each platform. Windows is most common among desktop computers which are most often using web-based applications. Connectivity and outages should be no problem. Mobile devices can access the game via wi-Fi and also cellular data connectivity.
4. **Security**: To make the game secure we will require a username and password login system for the app or downloaded game. Each player will have an identity and requirements to login such as an email address and phone number for account recovery. This will also allow for each player to be a unique person with their identity as their profile so they can play with friends. Using windows and coding in Java should be pretty secure because Java is a secure language, and windows implements many security features that are well rounded when using networks, such as windows firewall and a built-in web security center to protect you from viruses and hackers.

Resources

Microsoft Corp. (2018, May 31). About Memory Management. Retrieved October 11, 2020, from https://docs.microsoft.com/en-us/windows/win32/memory/about-memory-management?redirectedfrom=MSDN

The GIS Encyclopedia. (2018, July 3). Windows Memory Management. Retrieved October 11, 2020, from http://wiki.gis.com/wiki/index.php/Windows\_Memory\_Management#:~:text=Microsoft%20provides%20an%20resource%20describing,to%204%20gigabytes%20of%20memory.

Sulung, P. (2018, August 21). Windows Architecture — Top Design Inspiration Decoration Windows Architecture. Retrieved October 14, 2020, from https://medium.com/@putrasulung2108/windows-architecture-d2b022f136d3#:~:text=The%20architecture%20of%20Windows%20NT,symmetrical%20multi%20processor%20SMP%20based