

Draw It or Lose it

# **CS 230 Project Software Design Template**

Version 2.0

## 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 |
| --- | --- | --- | --- |
| 0.0 | 11/01/2020 | Andrew Vue | Evaluation of project |
| 1.0 | 11/13/2020 | Andrew Vue | Initial project set up |
| 2.0 | 12/13/2020 | Andrew Vue | Final recommendations |

**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 of the project are that the application needs to have access to a database of images to display for users. The initial issue is that the library of images needs to be large enough so that the same image will not be presented to the users within a given time frame and it also needs to be random. To do this we need to write code to pull the images from a database, have a randomizer set up and another layer to parse the image ID so that we don’t get a match for an image already used within the given time frame.

## [Design Constraints](#_2et92p0)

The initial design constraints I can see right off the bat are:

* The availability of an image library and the rights to use these images for the game/app
* Setting up an API to pull the images
* Creating ID for each individual image
* Connecting all of the different devices using this game

## [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 diagram below shows that different aspects of the game broken into chunks that will work together. The processes of creating each class and object to function as opposed to having one large file will make the game run smoother and faster. This allows for the developers to update and fix specific parts of the game as well. The Entity is what holds the player information and the Game, if there is a Team then players are a subclass of the Team. Game Services and the Game require Players or Teams to run.

****

## [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** | Objective-C  This is the most used language for Mac OS and is most robust language. Objective-C is a very powerful low level language that can work with any other language on the from side. | Many Linux servers use C, C++, PHP, and Java.  Linux is a relatively neutral system compared to Windows and Mac. Linux OS are some of the most popular servers, but not so much user devices. | Windows servers use C, C++, PHP, C#, and Java. The popularity of Windows devices is higher than Linux, but the servers are less popular than Linux servers. | The server-side languages don’t matter because that is dependent on which OS the servers are using. For the best connection I recommend Linux servers.  If Linux servers are used then the languages are C++, PHP, and Java. |
| **Client Side** | Objective-C and Swift should be used because these languages work best for Mac specific platforms. Swift was created to work specifically for Mac OS products. | Linux is not a popular OS among the average consumer so not many considerations are put into developing Linux specific programs. In most cases Linux can run most apps that run on Mac and Windows. | The development cost and time of using Windows is the development environment set up time being longer. Unlike Mac windows apps can be created using the same languages as its servers. | The development cost and time associated with mobile devices are that specific apps will need to be made and tested. This will take more time because unlike desktop computers using websites mobile apps will be used for optimal performance and experience. |
| **Development Tools** | When using IDEs it depends on the language, but in the case of Swift there have been many recommendations for Xcode, AppCode, and Atom. Objective-C works well with visual studio or any of the other IDEs. | The languages used are what dictates the IDEs, because different IDEs may have better features for specific languages. In this case this is dependent on the developer’s preference. | Visual studio is recommended for Windows because it is the IDE developed by windows and works with all the languages that will be used. | The programming languages and IDEs will depend on the OS of the mobile devices. For Apple/Mac products Swift must to be used. For Android, it can be C++, C# or Java.  There are OS neutral options, but these are new and not supported to the same degree as older languages. |

## 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.>*

The relevant programming languages and tools used to build this type of software will depend on the platform and operating system. To ensure the speed and quality of work I recommend having two teams one that focuses on the Mobile platforms and one team that focuses on the Desktop computer platforms. In these teams there will be sub teams; for mobile there will be a team for the iOS platform and Android platform. The Desktop team will not need to be broken down further because the game will be accessed via the browser/web so it will be standard across all Desktop operating systems if designed correctly.

1. ***Operating Systems Architectures****: <Describe the details of the chosen operating platform architectures.>*

The languages used and silicon used for different operating systems are different. Most of the hardware used today are coded in C or C++, but the software of these devices are different because of the targeted user experience. Mac OS and Windows have different user interfaces and experiences because they prioritize different things. This also creates competition in the market which is good because it promotes innovation. The differences in user experiences requires applications to function according to that OS. The demands required by Mac OS compared to Windows and vice versa were some of the reasons why certain apps only live in certain operating systems.

1. ***Storage Management****: <Identify an appropriate storage management system to be used with the recommended operating platform.*

The management of storage is important because it dictates the amount that can be stored and how quickly the information can be accessed. The current hardware available on the market for commercial use and average consumer use are impressive, but it is wiser to store the data of Draw It of Lose It in a cloud storage service. Storing the data in a third-party data center eliminates the upkeep of your own data server for a specific application or program. The overall cost will be lower compared to maintaining your own server.

1. ***Memory Management****: <Explain how the recommended operating platform uses memory management techniques for the Draw It or Lose It software.>*

The considerations and specific approaches needed to ensure memory is effectively managed in the software application for Draw it or Lose It are storage type, speed, consistency, and security.

When storing high-definition image files it takes a lot of memory and storage space especially since each image stored is x megabytes and when having xxx totaling to X gigabytes of storage space. There is also the requirement of having the proper hardware to store and transfer those data files without compromising speed, image quality, and accuracy. For the game to work properly the images need to be transferred at X MB/s so that the images pop up instantly. The file sharing and access also needs to be considered because there are differing operating systems and platforms that are using the application. The image sizes need to be the proper size for the platform it is being populated on.

1. ***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).>*

Distributed Systems and Networks are important in connecting different users on different devices because it ensures that regardless of the device type data can be shared and used. The problem with having different operating systems and device types is that sometimes the files and data from one to the other are not compatible and must be converted. Storing the data of the game Draw It or Lose It in a server will allow for different devices and device types to access the game.

1. ***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.>*

Security is important, but there are many levels of security in every application. The first line of security is in the coding practices used by the development team. The next line of security is writing actual security programs to detect and block breaches. The final line of defense of the built-in security features of servers used, and hardware. There is no such thing as 100% secure because there are always going to be people who try to breach your security measures, but maintenance and updates are the best approaches to keeping your application secure. Each operating platform has their own security features and will constantly update their operating systems to keep them secure. The best thing is to constantly be on the look out and audit the security of the application.