

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 | <04/11/2021> | <Sergio Ruiz> | Giving an evaluation when it comes to the OS for the client program Draw It or Lose It |

**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)

Our client “The Gaming Room” needs us to help develop their game called “Draw it or Lose it” into a web based app. Our duty is going to be to primarily assist in setting up the environment for the game. This environment will need to have one or more teams which will have multiple players given to them. The names, the team, and the player must be unique. There must also be a way to see if a name is already in use. All of this will be required to be put in place in a manner that is consistent with making a profitable app.

## [Design Constraints](#_2et92p0)

While working on this project, we will face some major and minor design constraints. One of the biggest constraints will be making sure we stay in budget and are on time. Our client is counting on our team to help them deliver a profitable app in a reasonable time frame. Our client also wants an app where only one instance of the game can exist in memory at a given time. We are going to have to present the app in a manner that is exactly what The Gaming Room envisioned. In order to make sure we deliver a product of the best quality, we must make sure we test the software and implement the feedback we receive from our client and the feedback from other users. We must find a way to include this in the budget and we must allot time for this.

## [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 shows the Object-oriented Programming principles which will be used for this app. Our team will be utilizing seven classes in order to implement the design how we want them. A parent class Entity will be used. This class will give base variables of Id and name for its child classes. The variables are going to have base getters for Id and name. They will be inherited by the child classes of Game, Team, and Player. By using the inheritance that is in the object-oriented programming method, we will save time. This is because this gives us all the variables and child classes we need. The Game Class will create a list for all teams created, also the methods to add a team name to a list. The Player class will just give a way of making a player instance. The GameService Class, is going to interact with other 4 classes using singleton pattern which will make sure that only one instance of a game exists at any one time. GameService is also going to include iterator patterns in order to make sure that the app can quickly iterate through the names so that there is only one active instance. For a SingletonTester class, will be used by the ProgramDriver class to test that our Singleton Class (GameService) is working how it is meant to be. The ProgramDriver class includes the main method so this means it will drive the app.

****

## [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** | Ease of use due to great UI. Easy accessibility and server configuration | Navigation is hard in this. Linux has command shell for server config. Is budget friendly. | Expensive but user friendly UI. Has a command prompt. Many software options for developers. | Specs vary from user to user. Inexpensive. Options are mainly Android OS and iOS. Making same game that works the same on all devices is challenging. |
| **Client Side** | Expensive for users. Amount of time needed is average. Must be accurate to work with Operating system. | Users need to be proficient. New users will need more time to work with Linux. Low cost for users | Not much time needed to learn, but cost is more than that of linux | Users need to give lot of time and have high skills to support mobile. Apps made for one platform might not always be easy to use in other platforms. |
| **Development Tools** | The IDEs for MacOS are Python, HTML, Javascript, CSS, etc. The software available already within Macs is Eclipse and Visual Studio. Online developing tools may also be used | IDEs are Ruby on Rails, HTML, Java and Python. Dev tools are Github, Repl.it, nodejs, Vistual Studio. | Languages are Python, HTML and C++. Dev tools are Eclipse HTML Repl.it and others. | Languages are: HTML, php, C++, Python. Tools for development are Github,Visual Studio, CMD, Repl.it. |

## 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**: An appropriate OS that was found to be able to run Draw It or Lose It efficiently would be Windows OS. Windows allows users to have ease of use and it also gives them access to many software packages. This will give them a great experience. Windows OS is also great for developers because it gives them a few software engineering abilities.
2. **Operating Systems Architectures**: Windows OS has a simple user interface. Apart from the many software packages available for developers, their command prompt lets devs configure server settings easily. It also allows these developers to work with many different programming languages
3. **Storage Management**: An appropriate storage management system that can be used with this OS would be cloud storage. This will allow for plenty of storage for developers
4. **Memory Management**: Windows uses memory by allowing up to 4 gb of memory to run applications smoothly. It uses physical memory and virtual address.
5. **Distributed Systems and Networks** There are common issues that can happen between distributed systems and networks. One of the issues is queuing repercussions. The others are routing and congestion problems. These issues can lead to failed components, lagging and poor performance overall.
6. **Security**: Windows OS gives clients the ability to help secure data going in and out of the system. Windows has built in spyware. It allows for virtual private networks that protects information from malicious behavior. Encryption can also be used for information such as passwords, credit card numbers etc. Windows also has Microsoft DirectAccess which makes sure that users are authenticated when connected to business networks.