

<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 | <06/15/2021> | <John Blackwell> | <Creating the ability for teams to play at the same time, creating specific usernames for player data and ensuring there is only one instance of the game at any time.> <Overview provided under evaluation section for the benefits of different software for the client-side, server-side, and different Dev tools.><Recommendation’s portion has my recommended software for the client, and why I have chosen these items. |

**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 Game Room wants to develop their game into a web-based application having the ability to have one or more teams involved in the game. Each team will have multiple players but for the system to recognize who is on which team we will have to develop distinct team and player ID’s. Because the Gaming Room only wants to always allow one instance of the game, we will have to develop unique identifiers for each instance of the game, the team, and players.

## [Design Constraints](#_2et92p0)

Design constraints will include developing this game into a web-based application that is available for multiple platform use, possibly cross platform use as well. There is the constraint of adhering to a budget and a deadline as well. We will need to satisfy the client by submitting a completed project that adheres to their standard, this means the game will need to be tested once completed and any bugs will need to be worked out after the app has been developed. Creating unique ID’s will be paramount to the success of this design which will allow the game to only have one instance as well as showing which players will be on their chosen teams.

## [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)

Below is a UML Class Diagram that includes the classes that will be implemented in the coding design of Draw It or Lose It. In the diagram we can see how the classes of the project interact with one another. In the parent class that is named entity we can see attributes that the children classes game, team, and player will all inherit. In the children’s classes we can see inheritance which makes the UML diagram easier to understand and represents the principles of object-oriented programming. We can see the game service class is using the singleton method which will allow us to code the single instance of the game as required from the gaming room. The game class will create the instance of the game, the team class will implement the team ID and allow a function to add a player, and the player class will add the player ID. Lastly, you will see a driver and a singleton tester which will make sure our code is functioning correctly.

****

## [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** | Good quality DUI and good interfacing, easily accessible workstation that makes server management seamless. One drawback is that Mac is expensive. | Like mac with easy server manipulation and accessibility, known to be cost effective. Linux also has a command shell that enables simple server config. | Large availability for Windows, trusted, easy server access and command prompts. Windows can become costly but easy sever interface and tools makes it worth it. | Can be sluggish in hosting web-based programs. Different OS will offer different challenges among the plethora of smart phones. However, it is inexpensive on the server side, it will be challenging to host Apple, Android, and WP OS’s. |
| **Client Side** | Mac is more expensive than other OS, user friendly and compatible with Mac OS, time efficient due to high compatibility. Mac takes a higher level of skill to operate. | Linux is not the most user friendly, hard to perfect so time will be longer, cost is relative, but skill is required. | More affordable than Mac, time is efficient due to the user-friendly access, it is very easy to use. | Excessive time may be required to support multiple OS from different devices. Cost is low and user skill required is minimal. |
| **Development Tools** | With Mac the most common development tools are available, JAVA, C++, Python and so on, many libraries already exist and development tools with be bountiful. | Similar coding languages will be available for Linux like C++, Java, there are additional languages such as Ruby on Rails and CSS. There are IDE’s that can be used and a wide variety of development tools available. | Same as the other two, common languages are used, Python, Java, HTML, et cetera. IDE editors like Visual Studios and Eclipse will be readily available to Windows as well. | Similar tools available but more challenging to develop on a smart phone device, OS platforms will run on Mac OS, Windows, Android, and Linux. Some Dev tools would be Visual Studio, Repl.it and power shell |

## 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**: For the gaming room and developing Draw It or Lose It I would recommend using Windows. Windows is a widely known platform that has many capabilities and offers a very user-friendly interface. This platform will cater to many different users and will be a cost-effective approach for the functionality you will receive.
2. **Operating Systems Architectures**: Windows has an easy to work GUI which will come in handy in game development, there is a command prompt for server configuration and will allow the developers to use their chosen language for the program. This flexibility gives us creative control of the app we are developing.
3. **Storage Management**: I would advise to use a cloud storage management. The cloud storage will allow the server to store data so the user does not have to, with this we will have a speedy application that is available to the user when ready.
4. **Memory Management**: To manage memory I would advise using cloud storage as well as hard drives, coupled with a memory trash bin in the code that will delete old data. The cloud storage can hold most of the important game features and the user hard drives can hold game saves and certain records data. Utilizing a trash bin will get rid of data that is no longer being used.
5. **Distributed Systems and Networks**: In order to have multiple platforms that can communicate with one another you will have to have servers that can handle that load. Issues may arise with server overload if the game is being played on a large scale. Using a distributed system can have component failures that will cause performance delays for our users but will outperform a single server.
6. **Security**: Windows security features will not allow outside devices to change system without administrative access. We can use windows because it is a secure system that also has auto encryption features if desired.