

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template** 1](#_Toc115077317)

[**Table of Contents 2**](#_Toc115077318)

[**Document Revision History 2**](#_Toc115077319)

[**Executive Summary 3**](#_Toc115077320)

[**Requirements 3**](#_Toc115077321)

[**Design Constraints 3**](#_Toc115077322)

[**System Architecture View 3**](#_Toc115077323)

[**Domain Model 3**](#_Toc115077324)

[**Evaluation 4**](#_Toc115077325)

[**Recommendations 5**](#_Toc115077326)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 01/22/23 | John Melton | Formulating initial plan and software requirements. |

## [Executive Summary](#_sbfa50wo7nsh)

Our client is Creative Technology Solutions. They have come to us to request assistance is setting up their software development environment. The goal of this project is to streamline their development of a web-based version of their game Draw It or Lose It.

## Requirements

* The game needs to be able to function inside of a website.
* The game will have the ability to have one or more teams involved.
* Each team will have multiple players assigned to it.
* Game and team names will be unique identifier associated with it. If a name is already in use then the game needs to alert the player and require them to enter a new name.
* Only one instance of the game will exist in memory at a time.

## [Design Constraints](#_2et92p0)

Above are the minimum requirements needed to satisfy the client in terms of functionality. Additionally the client wants to game to be able to be played on multiple operating systems. The potential operating systems that would be possible to use this application would be Linux, Windows, Apple, and ISO based phones. The game in be able to be played as a web based application and will need to incorporate web based languages in order to function correctly. The client currently has a functioning application for Android. It will be important to not negatively impact these current Android users while developing the game further.

## [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)

Looking at the UML Diagram we are able to get a better idea of the functionality and how the program will function. Below we can see that Entity in the superclass of Game, Team, and Player. We can interpret this since each of the classes inherits from Entity. Additionally we can see that we are utilizing aggregation though the has a relationship. This is shown since GameService has a Game, Game has a team, and team has a player. By demonstrating this relationship, we can see how each classes references and connect the other. This will help when developing the game through object-oriented programming.

**"The Gaming Room UML diagram. The top of the diagram is labeled as com dot gamingroom. Test boxes are placed in two layers. The first layer has three text boxes and the second layer has four of them. In the first layer, the 'ProgramDriver' textbox points to 'SingletonTester' textbox. The 'ProgramDriver' textbox contains the text 'asterisk main round brackets.' The 'SingletonTester' textbox contains the text 'asterisk testSingleton round brackets.' The arrow between these two text boxes are labeled 'open two angle brackets uses close two angle brackets'. In the second layer, there are 'GameService', 'Game', 'Team', and 'Player' text boxes. The 'GameService' textbox has texts arranged in two layers. The first layer contains games colon List open angle bracket Game close angle bracket, nextGamesId colon long, nextPlayer Id colon long, nextTeamId colon long, and service colon GameService. The second layer contains GameService round brackets, getinstance round brackets colon GameService, addGame open parenthesis name colon String close parenthesis colon Game, getGame open parenthesis id colon long close open parenthesis colon Game, getGame open open parenthesis name colon String close open parenthesis colon Game, getGameCount round brackets colon int, getNextPlayerID round brackets colon long, and getNextTeamId round brackets colon long. The 'GameService' box is connected with the 'Game' textbox with a line labeled 'zero dot dt dot asterisk'.  The 'Game' textbox also contains text in two layers. The first layers contains the text teams colon List open angle bracket Team close angle bracket. The second layer has Game open round bracket id colon long comma name colon String close parenthesis, addTeam open parenthesis name colon String close parenthesis Team, toString round brackets colon String. The 'Game' textbox is connected with the 'Team' textbox with a line labeled 'zero dot dt dot asterisk'. The 'Team' textbox also contains text in two layers. The first layers contains the text players colon List open angle bracket Player close angle bracket. The second layer has Team open parenthesis id colon long comma name colon String close parenthesis, addPlayer open parenthesis name colon String close parenthesis colon Player, and toString round brackets colon String. The 'Team' textbox is connected with the 'Player' textbox with a line labeled 'zero dot dt dot asterisk'. It contains the text Player open parenthesis id colon long comma name colon String close parenthesis and toString round brackets colon String. The 'Game', the 'Team, and the 'Player' boxes point to the 'Entity' textbox in first layer. The 'Entity' textbox contains text in two layers. The first layer has the text id colon long and name colon String. The second layer has Entity round brackets, Entity open parenthesis id colon long comma name colon String close parenthesis, getId round brackets colon long, getName round brackets colon String, toString round brackets colon String.**

## [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** | Typically, hardware and software work are integrated together well.  Has the advantage of being less targeted by malware and malicious software.  Disadvantage is cost and some software is unable to be used on MacOS that would be available with other Operating Systems | Linux is very efficient and typically allows for more customization with regard to background activities.  One major advantage is cost and is typically the most secure Operating System available.  Can be difficult to find applications and software needed for web hosting. | Windows is the most widely used Operating System and has the most resources available for web hosting.  A key advantage is useability and familiarity with the Operating system and how to perform functions. Also has the most third-party software for optimization and delivery.  A key disadvantage is that most malicious software is made for windows since it is the most prevalent Operating System. | Having a portable server creates issues that can be difficult to correct.  There many different options for hardware and each have advantages and disadvantages depending on the device.  Disadvantages are instability in resources at any given time. If the device receives a phone call or a swaps from Wi-Fi to cellular this could cause issues with consistency.  Additionally, security concerns are more prevalent since the device is not in a secure location or on a secure network as often as other options. |
| **Client Side** | For users unfamiliar with MacOS it can take a little time to become efficient with the OS. Typically costs more for the same hardware that is available in other systems. Depending on browser used the time to get working should be fairly minimal when compared to other options. | Linux is typically the most difficult to get started on and developing understanding on how to operate the OS. This is the cheapest since Linux requires minimal resources and is typically free to users. Depending on browser used the time to get working should be fairly minimal when compared to other options. | Most users are familiar with windows since it is the most commonly used OS in the market. Windows keys and software require a moderate investment but can be utilized for other purposes. Depending on browser used the time to get working should be fairly minimal when compared to other options. | Provides the most options for hardware and is the most portable when compared to other options. Minimal cost would be occurred since most clients would already have a mobile device to use before starting the project. When compared to the other options can be more problematic to use on different devices. |
| **Development Tools** | For MacOS Swift and Bash is generally the preferred languages. An additional option would be PowerShell 7. Preferred IDE would be Visual Studios Code. This IDE has countless plugins for customization and supports almost any language. For the web application Java, JavaScript, HTML, and CSS would be used when developing the web application. | Linux can use Bash, PowerShell 7, and Python are all good options. For IDEs we can use Visual Studio, Visual Studio Code, and Eclipse. For the web application Java, JavaScript, HTML, and CSS would be used when developing the web application. | Windows would typically rely on the C languages; However, PowerShell and PowerShell 7, Python, and Java are all commonly written on windows machines. Almost any IDE or software has a windows version available to use. Notably Visual Studio, Visual Studio Code, and Eclipse would all be good options for this project.  For the web application Java, JavaScript, HTML, and CSS would be used when developing the web application. | The languages used would greatly depend on the device used. Kotlin for Android and Swift for IOS would both be great options. Additional languages and resources would be needed for development that would not typically be needed for the other Operating Systems. For the web application Java, JavaScript, HTML, and CSS would be used when developing the web application. |

## 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**: I would recommend Windows for The Gaming Room. Windows would be the fastest Operating System for their current personal to get started on. Additionally, Windows meets all of the required minimum that their team to use and would incur the only a moderate cost on to begin.
2. **Operating Systems Architectures**: Windows used Graphical User Interfaces to deliver content to the user. By presenting data in this way the user is able to make selection and efficiently navigate the UI. Additionally with Windows the user is able to optimize their machine with an abundance of settings.
3. **Storage Management**: Windows offers many different ways to navigate their data storage including file paths and a GUI file explorer. This enables users to manage storage efficiently while maintaining an ease of use for the end user.
4. **Memory Management**: By programming the game in Java we can utilize the garbage collector to optimize memory. We will also only have one game instance at a time inside of memory which will further optimize memory on our system.
5. **Distributed Systems and Networks**: Since our game will be a web application it will be delivered via the internet. We will have multiple server available with different locations in mind to minimize latency between users. Additionally we will have servers to deliver and receive data from the user. For this we will be able to utilize SQL. Since we are using Microsoft products pairing our machines with SQL on windows servers will greatly improve efficiency between users.
6. **Security**: While security is a concern by keeping servers remote from the users, we will be able to greatly increase security for our userbase. Additionally, Windows Defender offers moderate protection for free with our windows machines to protect data locally from malicious entities.