

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 | 07-12-23 | Ethan Harper | 1st version Starting Document |
| 2.0 | 07-14-23 | Ethan Harper | 2nd version completing OS Evaluation |
| 3.0 | 07-27023 | Ethan Harper | 3rd Version Development for Different OS |

**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 software problem is that there is currently an existing application of the game *Draw It or Lose It* that exists as an android app. The development team needs to make a version of this app that can run in a web-based environment and have it function the same way as the android app.

## Requirements

* Have one or more teams involved.
* Have multiple players assigned to a team.
* Game and team names must be unique.
* Only one instance of the game can exist in memory at any given time.

## [Design Constraints](#_2et92p0)

The game must be developed with the ability to connect multiple users to multiple different teams at the same time and keep the data across all browsers in sync even if each user is using a different browser. The game must be able to check if each team name in the current game is unique so it must update all users with that data as soon as a name is selected and prevent the team from being able to select the name of the same characters. Only one instance of the current game can exist in system memory at the same time for each user. Must create identifiers for each game that prevents a game of the same identifier from being stored in memory.

## [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 below shows several classes that are essential for the game to run. Starting with the base GameService and branching out into Game, Team and Player classes that are inherited by one Entity class. This shows on of the OOP principles Inheritance, now all this program’s code is stored in a single class and the Entity class is the game session entity where all players, their teams and the games are run through. This shows abstraction as well, the main driver class can be used not just to test the singleton but to also run the entire game entity and the users will not directly access all the code as they will only interact with part of it without knowing how it all flows together. The project is encapsulated based on the state, it can make decisions on team names based on names that already exist in the entity as an example and can pull from Game, or player classes to get the information needed.

**"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** | Apple does offer server-based environments and having a MacOS server can benefit clients that are using a mac. The Mac server interface is easy to administrate and use but can be expensive to maintain. It is also not ideal for companies that use a lot of third-party programs. | One of the advantages of using a Linux server is that they are free and open source meaning that the deployment or use of tools is cheap or even free. Its ability to be highly customizable allows for higher security for a server as well. Some of the downsides include that Linux can be very difficult to learn and that some applications are not supported on Linux. | Windows has the advantage of supporting a lot of third-party applications and software. There are great development tools like MySQL and ASP.NET that are fully supported. Windows can be quite expensive. | For Android and iOS there are server options despite the fact that it may be impractical compared to that as a proper server computer. Oracle does offer server-side implementation for mobile. It can manage users, devices and be used for deployment of projects. Since the team at Creative Technology Solutions wants support for both iOS and Android this is a good option as Oracle offers development tools for both of these different mobile devices. iOS apps deploy using their app store instead of usual server deployment like Android, this is without the use of Oracle. |
| **Client Side** | Mostly a learning curve for the client. MAC is very easy use once the client has learnt it and takes no time at all. Since it is not opening source there would be high costs since it is an apple device | Costs would be low due to Linux being open source but Linux can be very difficult for new users to learn. Common tools for web development like JavaScript, Python and PHP are supported. | Windows is the most common operating system so time to learn would not be high but windows can be quite expensive. | Mobile can be flexible thus keeping costs low and since a majority of the population uses them there should be a small learning curve. There can be different Operating Systems based on the type of mobile device. |
| **Development Tools** | FlexiHub, Homebrew Xcode, Sublime Text, Visual Studio Code. Development tools like Xcode on Mac frequently require licensing and thus incur additional costs for the client trying to use these. | Sublime Text, VIM, Visual Studio Code, PyCharm for server developing. LINUX is historically open source and so many of there development tolls are as well. Some will still require licensing but many Development tools that work for Linux will also work for Windows. | Windows Software Development Kit (SDK) very useful. Allow for .NET development which is a unique way to develop web apps with minimal use of JavaScript allowing for better security. This can be used in Visual Studio. Many require licensing costs but some are free and open source and there are many alternatives when compared to MAC | Oracle, NoSQL |

## 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**: Windows is the OS I would recommend as the platform. Of the three Windows is the most in use and Windows Development tools especially using ASP.Net, Azure and their SDK makes developing for a Web Application simplified and gives many great options for in depth development. The ease of development and mass use of the OS between Personal and Work on computers would make this a great fit and outweigh the costs of Windows. Since we are attempting to develop cross platform, this would be great as well as Android and Linux systems can use some of the same tools and capabilities as Windows allowing for easier cross platform development while MAC OS requires many of its own unique requirements.
2. **Operating Systems Architectures**: Most operating systems have similar architectures. This would be the hardware, kernel, and the shell. Windows allows the ability to access the OS shell through command prompt, shell or bash shell that can access other services and the OS. Using IIS can manage server systems and access programs that communicate with the OS.
3. **Storage Management**: Owning a Windows cloud server can be very expensive but the wide range of simple configuration and depth for types of storage and security of that storage makes the costs worth it. Cloud storage is the recommended way to store data for the game as it is an online Web Application, and this allows for fast user speeds in many different locations across the world. Blobs, Caches and other storage units offered by Microsoft allow for many options here as well and can be utilized using the cloud shell and local command prompt.
4. **Memory Management**: For Windows you have both the option of Virtual and Physical memory. Virtual memory makes large scale apps more practical. This also has the added advantage of memory protection and the ability to work with and extend physical memory. Physical memory like increasing the amount of RAM (Radom Access Memory) on a physical server to run the games is also a possibility. This can be very expensive and require even more frequent upgrades then having virtual memory that you can easily change the payment plan on.
5. **Distributed Systems and Networks**: Sometimes outages caused by physical occurrences can cause disruptions that are unavoidable. Luckily Windows allows for backup servers and the ability to add and purchase greater strength networks that can spread out across several countries. This would allow Networks to reach more and have closer local networks so that users do not suffer connectivity issues.
6. **Security**: Windows offers many safety features by default including Windows Defender antivirus program. There are also many VPN services out that can be added for a protection of the user information. Being able to track and log potential security threats is also a feature that Windows offers. Also setting up multi-factor authentication on all computer systems and resources on a Windows machine to prevent someone from logging in with another partially complete data. This can be a sub-part of RBAC (Role Based Access Controls) and lock users based on assigned roles and can allow temporary roles to be applied. Setting up physical security on work computers like locking doors, requiring log in passwords and hiring trust worthy individuals who won’t steal would apply to all systems.