

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 | 01/20/2022 | Brandan Rodriguez | Creating the Executive Summary, Design Constraints,  and Domain Model. |
| 1.1 | 02/18/2022 | Brandan  Rodriguez | Completing the Evaluation |
| 1.2 | 02/27/2022 | Brandan  Rodriguez | Completing Recommendation |

## [Executive Summary](#_sbfa50wo7nsh)

The Client “The Gaming Room” objective is to develop a web-based game called “Draw it or Lose it” and that that game is compatible to multiple operating systems. The is based on a 1980’s game “Win,Lose, or Draw”, where team competes of what is being is being drawn. However, in this game an image would be slowly rendered within 30 seconds for completion.

Rules of the game:

1. Team given one minute to guess correctly
2. If team does not successfully guess, then opposing team is given 15 seconds for a chance
3. Game consists of 4 rounds

To make this game available we would be required to host web server that would provide the game application available to users in the web and develop the game application to be compatible for cross-platform.

## [Design Constraints](#_2et92p0)

* Two or more teams
* Players in the team
* Game and team names must be unique
* Render Image to reveal puzzle
* A timer to time the tasks
* Only one instance of game
* Game must be compatible for multiple platforms

The above reference are requirements for game software. The goal is that the game can run using different operating systems such as Windows, Linux, Android, IOS, Mac or Chrome in the web. The difference in the OS varies on the programming language that the game is written and the web support. Hardware of the server system should be efficient to support the web game and end user, such as the CPU, Ram, storage, and graphics. A stable internet connection bandwidth is needed to maintain the online connection. The type of OS used for server is needed since each have their pros and cons for the functionality. A firewall should be invested for the server to maintain max performance and prevent hacks or malware infecting the server. This would maintain the game to perform in the web and end-users.

## [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)

We can see that Entity is a super class or a base class because Game, Team, and player inherits information from Entity. This shows the use of iterator pattern to improve maintainability. For example Game, Team, and player have the same method “toString()” from Entity. We also see aggregation relationships “has a “type. A GameService has a Game. A Game has a Team. A Team has a Player. Each relationship listed has 0 to many. This is an instance of a class and has a reference has a reference to an instance to another class. player, Game has a team, and GameService has a Game. This means that “has a” is an instance of one class and has a reference to an instance to another class. The classes have attributes in the second box and behavior in the third box. Accessor and mutators used in the class such as “add” or “get” methods. SingletonTester class verifies that the GameService class has only instance of the class and must be externally accessible by using the singleton pattern.

**"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)

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | The accessibility to configure server by using terminal commands.Can be used to webhost for game application.  Advantage is availability upgrades to create the webhost.  Downside not that popular for hosting web services. | Same goes as the mac, but Linux is cost friendly.  Popular to host web servers.  Known for its secure security.  Disadvantage finding the application that support the webhosting needed | Large access to software than other OS.  Widely used and close to other platforms.  Advantages high access resources requirements, and user easement.  Disadvantages the amount of rebooting needed and cost bit higher than Linux. | It is portable but recommend staying stationary.  Popular and portable.It has higher wider reach and affordable.  The downside is that it insufficient hardware to keep up with large demand of requests |
| **Client Side** | Apple products are pricier among the four. Sufficient expertise normal use and time. The requirement is what application needed to provide cross platforms. | Linux low cost. The expertise is skilled, and time required. The requirement is what application needed to provide cross platforms | Cost can be high as mac. Expertise needed is minimum and time. The requirement is what application needed to provide cross platforms. | Provide large accessibility to clients and developers to check updates if the device has internet service. require time for two and expertise. |
| **Development Tools** | Swift is used various for languages and while using notepad++, it amplifies the usage of all programming languages. Mac can run all languages. It can be Java, Python, PHP, and Ruby. Also, for web development HTML/CSS/JavaScript. | Linux can use visual studio, eclipse, and notepad++. Provides. Linux It can be Java, Python, PHP, and Ruby. Also, for web development HTML/CSS/JavaScript. | Easier to use in Window than Linux, but also can the same IDE as Linux. It can be Java, Python, PHP, and Ruby. Also, for web development HTML/CSS/JavaScript. | You are able create variety of apps using android and swift. The languages used for both platforms can run all three platforms. I can be Java, Python, PHP, and Ruby. Also, for web development HTML/CSS/JavaScript. |

## Recommendations

1. **Operating Platform**: I would prefer the Windows operating system for The Gaming Room to start off based off on the qualifications of the server side, client side, and the development tools provides.
2. **Operating Systems Architectures**: The Windows provides various services to show graphical user interface and services provided in OS such as messaging, webservices, and networking. This can provide for the user of use of the server. Windows is available in both 32- and 64-bit architecture with versions like pro, enterprise, education, server, etc. The difference between the architecture will determine the amount of memory it capable of addressing and the OS functionality.
3. **Storage Management**: In the windows 10, there is access to the saved data like the storage data. This can allow the user to manage and view the stored data. The feature categorizes the information and view the percentage of consumption. This also provides the use of a cloud storage like google drive and physical back-up in case of a drive failure. Also, security can be added to the storages with the use of encryption to secure data. SSD would be the choice for speed and budget.
4. **Memory Management**: The game would consist of a library of pictures, so the memory allocation allows access for easy storage. This provides the project to be put together in a more secure spot provided by OS. 64-bit Microsoft Windows has its own user-mode virtual address space for several terabytes. Polymorphism is in code which protects a process from getting access by another process that can potentially be corrupted.
5. **Distributed Systems and Networks**: We would need a cross-platform game so it can be run into different operator systems. I have found that the use of Drop wizard providing REST API can provide this beneficiary to be hosted and adaptable for cross-platform in web. Window OS has capabilities to manage hardware resources like network cards and local services to remote clients. This provides the functionality to support the network game.
6. **Security**: Windows already has a built-in security Microsoft Defender Antivirus for real-time protection and system scans for protection. This can prevent in some degree against malware, virus, and security-threats. Updates like the OS and applications are provided to help the users from new threats like bugs or exploitation.