

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 | 01/22/2023 | Chris Wong | Develop web-based game on multiple platforms |

**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 Gaming Room is interested in developing a web-based game named “Draw It or Lose it” that can be run on multiple platforms. The objective of the game involves multiple teams with a few members to spend four one-minute rounds guessing what a slowly rendered image is that gets pulled from a collection. If nothing is guessed before time expires, the remaining teams have fifteen seconds to correctly solve the puzzle.

## Requirements

< Please note: While this section is not being assessed, it will support your outline of the design constraints below. In your summary, identify each of the client’s business and technical requirements in a clear and concise manner.>

## [Design Constraints](#_2et92p0)

* *The game has one or more teams playing*
* *Each team has multiple members*
* *Game and team names are unique, to avoid naming confusion.*
* *Only one instance of the game can exist at any given time.*
* *Multi-platform support*

*Given the aspects of the game that need to be included, there are also the more software-related constraints such as developing and supporting several different platforms such as Windows, MacOS, Linux, and mobile devices as well. Developing tools such as the rewriting of existing code to both support and inherit from different languages to strengthen compatibility.*

## [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 team, player, and game classes have a relationship with the newly developed Entity class. The relationship is through inheritance of attributes they all share, such as name and id. This UML demonstrates that those relationships make Entity to be a super class.

The relationship between teams and players is a "has a" relationship. A team has players, while a game has teams and game service has games. In UML diagram this is represented by aggregation or a “has-a” relationship which means that an instance of one class has a reference to an instance of another class.

**"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** | It has adjustable commands for both access and configuring of the server. Mac does have MacOS server available, but it was discontinued recently. So future reliability is something to keep in mind. | Easily configurable, large amount of documentation, and utilizes a dynamic module system used to add features. It has a tough time handling substantial amounts of requests. | Windows server is reliable, cost-effective, comes standard with core network services like user authentication, DNS, DHCP, and secure file sharing. It also supports a wide variety of 3rd-party programs. | Hosting a server on a mobile device would prove challenging due to the potential lack of processing power. It is also not advisable to have something like a portable server. |
| **Client Side** | Time to establish the server would depend on the expertise of the user. The cost would be similar to a Windows-based server. The development of the application is expected to be compatible with various web browser systems and mobile devices. | Linux, compared to Mac and Windows might take longer due to its open-source nature, and it being one of the least popular for server solutions. Finding someone with the time and expertise to establish the server would be essential. Thankfully, due to that very same nature it would also probably take the least amount of money to establish and maintain. | It is user-friendly, and the time required to learn to use it is low. Cost is low considering its potential. It also can scale both in software and hardware. | Allows developers to view updates from any location.  Implementation is slightly more complicated due to having to make it compatible with several different mobile devices. |
| **Development Tools** | Using Swift (Apples Language) to write applications for Mac would be ideal. But C++ and C can also be used as well. For libraries, you could use other programs such as Visual Studio or Eclipse. If not, several IDEs, such as Atom for Swift, are also available. | Due to Linux’s open approach, most popular languages are supported, such as C/C++/C#, Python, Java, and even Swift. Using Visual Studio, Sublime Text or Atom are all solid choices depending on the programming language needed to be developed. | Visual Studio is one of the more popular options due to its multi-language support and is an ideal to create a development environment in the cloud. There is also Eclipse which is great for multi-platform development and supports several different languages as well. | IOS devices would follow similarly to Mac development, utilizing Apple's IDE Swift and Xcode together. You could use Android Studio, the official IDE for Android devices for that platform. Eclipse and Visual Studio with Xamarin could be utilized as well. |

## 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 the client to first start with development on Windows, as it has a wide variety of software and IDEs, is not as expensive as Apple devices, and lastly requires less time and expertise compared to Linux.
2. **Operating Systems Architectures**: The Windows architecture allows applications to take advantage of the platform's kernel processes without directly affecting them. In other words, applications use features of Windows to set up the GUI/windows that run the application, access memory, and other critical processes without inadvertently affecting the processes that run the operating platform.
3. **Storage Management**: Windows has disk management and storage sense built into the operating platform itself, including a built-in disk cleanup tool. Disk management is a Windows utility primarily used for advanced storage tasks. On the other hand, the disk cleanup and storage sense features are mainly designed to help you manage system storage and improve system performance by removing unnecessary files that occupy disk space.
4. **Memory Management**: Windows provides memory management features such as physical and virtual address spaces that support up to four gigabytes of memory and ways to run applications quickly.
5. **Distributed Systems and Networks**: Here, we will use a client-server distribution system where each client program will rely on the single server application for our game so that each client application can be tailored to the capabilities of that client system. The success of this game depends on numerous clients connecting to a single server and playing the same game at the same time; therefore, a robust server network would also be required.
6. **Security**: Data or sensitive information breaches can be avoided with the aid of provided security layers. This feature also blocks damaging attacks while enhancing overall security capabilities. Virtual machines with shields are yet another capability included with Windows Server OS. Using VMs, host administrators and managers may stop unwanted access to protected data. The management of which apps execute on the computer is ensured by Windows Defender Application Control. There are no additional hardware or software requirements for this capability. Users of the Windows Server OS also have built-in defenses against attacks that cause memory corruption. Windows Defender, which detects and disables known malware, is another feature of Windows server. Active Directory network traffic and SIEM data are both used by the advanced threat analytics functionality to find and alert users to potential risks.