

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.2

## 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 6**

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 03/18/2023 | Jeffrey Sparks | *Changes have been made to the cover page, the document revision history, the executive summary, design constraints, system architecture view, domain model, and recommendation.* |
| 1.1 | 03/09/2023 | Jeffrey Sparks | *Changes have been made to the development requirements table within the Evaluation section (p.4)* |
| 1.2 | 04/13/2023 | Jeffrey Sparks | *Changes have been made to the following sections:*   * *Recommendations* |

**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 project’s goal is to develop a web-based game that serves multiple platforms/Operating Systems. It is based on the popular game ‘Draw It or Lose It’. The idea of this application is to have multiple teams consisting of a variety of players, engaging in four rounds, lasting a minute per round. During each respective teams turn, The application renders images from a library pool of images. By the 30 second mark, the image is fully rendered. If the team does not guess by the expiration time, each member on the other team has the opportunity to guess, with a 15 second time limit. Currently, ‘Draw it or Lose it’ is only available via Android app.

## Requirements

* *A 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 must be unique to allow users to check whether a name is in use when choosing a team name.*
* *Only one instance of the game can exist in memory at any given time.*

## [Design Constraints](#_2et92p0)

A few constraints that will exist within this project exist in the form of making a consistent experience within the web-based application and the mobile application. The goal is to reach a wider audience, but to do so successfully. This will include further training in maintaining the web program after completion. Budget can also be a big concern due to the complexity of creating a consistent experience amongst all users, because not all web browsers are made equally.

## [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 shows that we created a package containing the classes for Entity, Team, GameService, Game, Player, Program Driver, and SingletonTester. We developed the Entity class as a base class that has an inheritance relationship with the Game, Team, and Player classes. There is an associative relationship between GameService, Game, Team, and Player classes. Each instance of GameService “has-a” (represented by 0…\*) instance of the Game class, which in turn has an instance of the Team class, which in turn has an instance of the player class. The dependency on the instances throughout the program creates the structure needed to iterate through lists that exist within the Team and Player classes that help enact a requirement from the Gaming Room to ensure that there is only one instance of each team and each player during any given game. The program driver class is shown to use the SingletonTester class to ensure that there is only one instance of the GameService at any time.

**"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** | Mac OS has easy accessibility and server configurability with an easy to use GUI (Graphical user Interface), featuring flexible terminal commands. Available only on Apple-products. | Linux is cost-efficient, mostly free, and mostly open source. It can be difficult to navigate based on user-experience. It has a command shell for simple server configurations. | Licensing fees can be costly, making it more expensive than Linux, yet is more accessible than macOS. User friendly GUI (Graphical User Interface) and contains a command prompt. Accessible and works with server services such as Microsoft Azure. | Mobile device specifications vary model to model, by extension person to person. Hardware can effect the ability to run software as most applications are not cross-platform compatible. |
| **Client Side** | MacOS is made by Apple. It controls about 10% market control on the desktop computer. This makes it less desirable for external attacks and adds a feature of safety to its users. It has a sleek design that connects seamlessly to other Apple-branded products. Though when it comes to hardware, Apple tends to be a bit rigid, lacking customizability. Because of the lack of Graphics Memory, Mac OS isn’t good for gaming, and can be considered expensive since it is exclusive to Apple-branded products. | Linux is open-source, stable, reliable, highly cost effective and customizable. Linux does have a lack of software and hardware availability, commercial support, a steep learning curve, and compatibility issues. It is not necessarily the most user-friendly to those not of an IT/Computer background, and has less exposure when compared to other major operating systems. | Windows is the most widely used personal computer operating system. It is user-friendly, has great software compatibility and hardware compatibility with customization possibilities. Most games released for Linux or mac are also released for Windows. Most games developed for older versions will still work on newer operating systems. Windows can be expensive to operate due to licensing fees. Windows can also be victim to security issues due to such a wide user base. Windows also has a very demanding drain on system resources when compared to its counterparts. Software fees and may be costly, and frequent automated updates on windows may cause temporary issues or bugs. | Mobile devices have weaker hardware than what traditional PCs contain. They are highly convenient and multipurpose with features for communications, location tracking, NFC payment options, cameras, and various others. Mobile devices are increasingly common and are great for purposes of entertainment. Cost can vary based on specific devices. Can offer seamless transitionary experiences with other operating systems (such as iOS seamless compatibility with macOS). |
| **Development Tools** | Some developer tools that are available for mac OS:   * PyCharm * Xcode * CLion * Eclipse * Visual Studio * RubyMine * WebStorm * Wing IDE * Apple Developer Tools * Android SDK * Komodo IDE   Some programming languages include:   * C++ * C * Swift * Objective C * Python * AppleScript * Ruby * Java * -HTML * CSS | Some IDE developer tools available for Linux:   * Eclipse * Visual Studio * Pycharm * CLion * Komodo Edit * Anjuta * Android SDK * Oracle SQL Developer   Some programming languages include:  -Python  -C++  -Perl  -Java  -Google Go  -SQL  -HTML  -CSS | Some IDE developer tools available for windows include:   * PyCharm * Visual Studio * Eclipse * Datagrip * Flowfinity * PhpStorm * IntelliJ Idea * Net Beans   Some programming languages include:  -C++  -C  -Swift  -Python  -Java  -JavaScript  -HTML  -CSS | Some IDE developer tools available for mobile devices include:   * Android Studio * Qt IDE * Xcode * Eclipse IDE * JetBrains Rider * Visual Studio * Droid Script * Apple Developer Tools   Some programming languages include:  -Swift  -C  -C++  -Objective C  -Java  -Python  -JavaScript  -XML  -HTML  -CSS |

## 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**: With its sheer number of users, I would recommend Windows, the most widely used and available operating system.
2. **Operating Systems Architectures**: The Windows API (Application Programming Interface) is the programming Interface of the Windows OS. It provides services used by all Windows based applications to enable applications to provide a Graphical User interface (GUI), access system resources, incorporate audio and much more. Some major categories of Windows API functions include:

* Base Services
* Component Services
* Graphics & Multimedia Messaging
* Networking and Web Services.

1. **Storage Management**: Within Windows 10, one of the modern iterations of the Windows OS, is a feature called “Storage Sense”. The purpose of this feature is to allow the user to allow windows to automatically free up drive space when items that are no longer needed are cluttering the drive space, such as temporary files and items in your recycle bin that you may forget about.
2. **Memory Management**: <Windows uses techniques such as Swapping (By the use of the swap-in, or moving the program from the hard disk to the RAM, and swap-out, or moving the program from the RAM to the hard disk.), and memory compaction ( Copying all pages that are not in use to one large area, writing all pages that are currently being used into newly freed space, reducing or eliminating fragmentation of the data and make it easier for programs to allocate memory) .
3. **Distributed Systems and Networks**: By using something like the HTTP (Hypertext Transfer Protocol) you can load web pages using hyperlinks. It acts as an application layer protocol designed to transfer information between networked devices and runs on top of other layers of the network protocol stack. Using an HTTP allows information between clients and servers to be transferred. HTTP relies on the TCP reliability. It uses a multiplexing technique which sends multiple requests through a single TCP connection, making it more efficient.
4. **Security**: Since Windows is the most widely used operating platform, security can be a concern. Windows does provide a basic anti-virus and firewall system to help combat computer viruses and other malwares. If you choose to not use Windows pre-installed software, a third-party anti-virus and anti-malware software would be recommended for security.