

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 | 11/11/2023 | Mitchell Fontaine | Provided executive summary and design constraints.  Added Entity Class and updated existing classes. |
| 1.1 | 11/25/2023 | Mitchell Fontaine | Added Evaluation section and provided relevant information to the client. |
| 1.2 | 12/06/2023 | Mitchell Fontaine | Provided Recommendations section based on the needs of the client. |

**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)

<Write a summary to introduce the software design problem and present a solution. Be sure to provide the client with any critical information they must know in order to proceed with the process you are proposing.>

## Requirements

Our client, The Gaming Room, has an existing game on the android app store called “Draw It, or Lose It”. They wish to expand from their mobile platform to a web-based platform. This iteration of the game will generate images from an online library of stock images instead of users drawing on an easel. The staff at The Gaming Room does not know how to set up the environment so, in order to facilitate the development of the web-based version of the gaming app, they will need our help in streamlining the development.

## [Design Constraints](#_2et92p0)

1. Must be written in Java programming language.
2. Existing APIs must be reviewed.
3. Must run on multiple platforms.
4. Only one instance of a game can run at a time.
5. Each team must be able to have multiple players.
6. Game and team names need to be unique and allow for checking if a name exists in the system.

## [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 provided in the software design document consists of seven classes. Our new Class, Entity, has a relationship with the Classes Game, Team, and Player. This is an example of multiple inheritance. The Class known as GameService has a direct association with the three classes of Game, Team, and Player while also demonstrating multiplicity. Then we Class Singleton Tester, which is constantly making sure that our program only has a single instance of the game running in a single instance. This Class has an association with the Class ProgramDriver, which contains our Main method.

**"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 has a simple server-based deployment with many products available as well as great tech support options. It has a proven track record of being security minded and there is plenty of documentation available for their software. With Mac you are limited to MacOS and few choices for hardware. This choice is also the most expensive available. | Linux is available as open source and has little hardware requirements. The platform is very stable and is constantly being updated by the open-source community against threats. This option is the most cost effective but does not have as many software options available as the others. | Windows also has a simple server-based deployment with an enormous range of hardware and software options. Windows also has some of the best technical support available and it receives updates to its services routinely due to the size of its userbase. Windows does have a less secure platform when compared to Mac and Linux and has fewer customization options. | Mobile devices aren't as extensive as the other options but do have the ability to manage server-side calls and queries to the database while keeping the code tucked away from the user. With this option, you would need to have physical or cloud-based servers available for it to integrate with. |
| **Client Side** | On the client-side Mac has many tools available for developers to use and allows for them to test software across multiple different browsers. The development and deployment time on Mac is fairly fast but not the fastest on this list. On the client-side Mac does require MacOS and hardware. | Client-side Linux has a large amount of development tools and supports virtually all available web browsers due to it being open source. The development and deployment time for Linux is also extremely fast compared to Mac and is very cost-effective due to it being open source. | Windows on the client-side also has a large number of available tools for developers and is supported on multiple browsers due to it supporting a large user base. The development and deployment time is very quick like Linux, but it does have issues with MacOS and has high costs. | On the client-side mobile devices allow for easy but slightly slower development time on Android and its costs are relatively low. There are some issues with testing on different browsers though. |
| **Development Tools** | Mac has the ability to support many programming languages like Java, JavaScript, HTML, CSS, Python, C++, C#, PHP, Ruby, and C but it also has its own unique language in Swift. Some tools and IDE’s available for Mac are Xcode, NetBeans, Eclipse, PyCharm, Visual Studio Code, Homebrew, and iTerm2. You can run Windows and Linux via Virtual Machines as well. The licensing costs on Mac are fairly cheap for development. | Linux can also support many programming languages such as C, C++, C#, Python, Java, JavaScript, HTML, CSS, PHP, Ruby, and Perl. Some tools and IDE’s it can utilize are Visual Studio Code, Eclipse, JetBrains, NetBeans, Sublime Text, Bluefish, Git, Geany, shell prompt, and the terminal. You can also use a virtual machine to run Windows and MacOS. There is no licensing cost for Linux because it is open source. | Windows has the ability to support all of the programming languages that the others can in addition to MATLAB, PowerShell, Kotlin, Nix, and Delphi. The development tools and IDEs available for Windows are vast and have exclusive use of the Windows Community Toolkit. You can also run a virtual machine to run Linux and MacOS. Windows licensing is free if you are already using their deployment services. | Mobile devices are much more limited in programming languages available with only Java, Kotlin, Swift, and Python. The development tools and IDEs available are Android Studio, Xcode, Xamarin, IntelliJ, Visual Studio, Flutter, and TestFlight. Mobile devices have a longer development time and have a more difficult time testing in other environments. You have to pay for licensing through Apple and Mac but not if using Android. |

## 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**: Based on the needs of The Gaming Room to expand their game Draw It or Lose It to other computing environments, I would recommend they go with Windows as their operating platform. Windows is very popular and easy to use. It also allows the use of other computing environments via virtual machines. They also have the best customer support of any operating platform.
2. **Operating Systems Architectures**: Windows architecture is a layered design featuring two main components. Those components are User mode and Kernel mode. User mode is restricted to running drivers and applications and also includes the Windows API, which enables applications to run a GUI and many other front-facing features. The Kernel is where the key components of the operating system are located. It deals with networking, memory management, hardware management, and other low-level processes.
3. **Storage Management**: The current Windows operating system has a feature called storage sense that enables you to keep track of how much storage you have left. Windows cloud-based platform Azure is the best choice for handling storage needs. Being cloud-based allows The Gaming Room to scale its storage to its current needs.
4. **Memory Management**: The Windows operating platform uses the memory manager feature that handles the allocation and deallocation of memory. “The memory manager implements virtual memory, provides a core set of services such as memory mapped files, copy-on-write memory, large memory support, and underlying support for the cache manager” (Alvin Ashcraft, 2021).
5. **Distributed Systems and Networks**: Windows supports cross-platform communication applications that can be integrated into the game. Microsoft Azure can store all user information in a database and enable said data to be used by players as needed. Azure also can scale to client needs so you don’t have to worry about it in the future.
6. **Security**: Windows offers some great security features. It has Secure Boot, Trusted Boot, and Measured Boot which all help prevent malware and corrupted components from loading once the boot process starts. Windows 11 offers other features like virtualization-based security and hypervisor protected code integrity. Microsoft Azure has built-in data security and encryption, an identity and user access management system, network and system security, as well as intelligent safeguards to protect from new and emerging threats.

References

Alvinashcraft. (2021, January 7). Memory Management (Memory Management) - Win32 apps. Microsoft Learn. <https://learn.microsoft.com/en-us/windows/win32/memory/memory-management>