

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 | 07/18/2024 | Steven Copeland-Helzer | Added an executive summary and design constraints. Filled in info for requirements to aid in the design constraints section, though this section is not final. Lastly, I reviewed and described the UML diagram of the application. |
| 1.1 | 07/29/2024 | Steven Copeland-Helzer | Added the server side, client side, and development tools fields in the development requirements table. |
| 1.2 | 08/16/2024 | Steven Copeland-Helzer | Completed the Recommendations section. |

**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)

Our client, The Gaming Room, wants to develop a web-based version of their Android app Draw It or Lose it. In doing so their game will increase its marketability by being available to an entirely new audience.

## Requirements

From what they have outlined - the 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, and only one instance of the game can exist in memory at any given time. This can be accomplished by creating unique identifiers for each instance of a game, team, or player.

## [Design Constraints](#_2et92p0)

We need a team who can take the existing code used in their Android app and work to translate that into code for the web-based version. From the requirements above, the unique game and team name requirements can be accomplished by creating unique identifiers for each instance of a game, team, or player.

## [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)

From the UML below, Entity is the parent class with Game, Team, and Player as its child classes. Therefore, these three child classes inherit the information provided in the Entity parent class. Along the bottom we see four classes all connected through aggregation. From GameService and Game, Game and Team, then lastly Team and Player. These classes all refer to an instance that occurs in the other. ProgramDriver holds the main argument used to run the application and uses the SingletonTester class. These last two classes are not connected to any of the previously mentioned classes.

**"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** | Easily connects to iOS providing ease of use between the two. Top notch security features. Limited to the Macintosh line of products. | More cost effective, open-source OS. Currently the most used server option. | On the server side it is more expensive, though it can be used on many different product lines. | Either Android or iOS. IOS is limited similarly to macOS, Android is more flexible. Both need to use cloud servers. |
| **Client Side** | Expensive to buy into, though they tend to last longer. Easy to use interfaces with second to none security features. | Interfaces are less user friendly with more difficult to navigate UI. Though it is open source it is still very secure since many developers add security measures to the OS. | Extremely user friendly. Many options to choose from with many different price points. Having security set up on your device is highly recommended. | Easy to use on both iOS and Android and both come with their individual pros and cons. Full features seen on PC’s are generally unavailable on these devices. |
| **Development Tools** | Swift is very popular but still uses Java, JavaScript, and Python. Has several IDEs available including Visual Studio and PyCharm. | C, C++, Java< and Python are all popular languages. Eclipse, VS code, and PyCharm are all available IDEs. | Mostly C and C++ as well as Java and Python. Visual Studio and VS code are the main IDEs used though PyCharm, Eclipse and more are also available. | Swift and Java are the primary languages used. Xcode is used for iOS where Android studio is for 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**: To expand Draw It or Lose It to different computing environments, we advise using Windows over all other operating systems. These factors make this platform a serious contender:

- Works with the most recent Android build. Install Android Studio or select a cross-platform program such as Xamarin, React, or Cordova to set up your Windows development environment.

- Windows makes it simple for developers to create, develop, and distribute software for Windows PCs by utilizing a number of Microsoft products.

- Numerous cmd/powershell/Ubuntu emulators are available for Windows, enabling testing on all platforms.

- Most widely used operating system and largest user base for game development.

1. **Operating Systems Architectures**: User mode and Kernel mode are two distinct operating system components in Windows. The majority of the user's interactions are impacted by user mode processes, which are user facing. Kernel mode deals with inputs and outputs, memory management, networking, hardware management, and routines. It is more low level and beneath the hood. Windows stores data in a directory structure. Windows also enables hardware modularity and multiprocessing, enabling system customization.
2. **Storage Management**: We strongly advise using Microsoft Azure for storage because to their affordable costs, first-rate customer service, and ongoing improvements and support. Azure also provides the following extra features:

- To take advantage of cloud storage instances, Docker containers can be installed in the Azure Cloud computing environment.

- Cloud-based storage makes it simple to scale up or down based on user volume. The storage could be expanded during the initial launch when it is anticipated that there would be a lot of users.

- Azure offers the Azure File System, Azure Storage Containers, and Azure Blob Storage as storage alternatives.

1. **Memory Management**: The most recent release of the Windows operating system, Windows 10, has once again enhanced memory management to enable quicker and more effective loading from memory. The main methods for accomplishing this are disc paging and demand paging, which serves as an addition to the computer's RAM and physical memory. Disc paging achieves this by setting aside space on the hard drive for additional RAM. Demand paging divides processes into smaller tasks that are only put into memory when a job is urgently needed for processing. The virtual memory address space is fully accessible to each process in Windows 10, which is more than adequate for the program.
2. **Distributed Systems and Networks**: Another accessibility while working with networks and dispersed systems is another reason we advise choosing them as your cloud service provider. With cloud-based email alerts, Azure App insights Logging, and available monitoring tools, Azure guarantees optimum uptime. When scaling up to allow 1000 simultaneous games with four players each, this will be crucial.

- By shifting the network load to Azure, you can put more of your attention on the application's features.

1. **Security**: Azure makes the logistics of user information and personal data security simple. The connected devices will establish a connection with the "Azure App Service" that is active on an App Service Plan and uses Azure Active Directory to authenticate users. - IP setups for whitelist-specific access to resources (like player or personal information) or the entire app are some additional features made available by Azure.

- A cloud storage solution that includes a VPN for increased security

- A database might be password-protected, only allow access from IP Whitelists, and demand SSL communication to safeguard user data.

- Possibilities to obscure user data to safeguard private information in the event of a breach.

Citations:

<https://azure.microsoft.com/en-us/services/storage/files/#features>

<https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-database-security/?ef_id=037eecb058b31669a99b888c6d034e7a:G:s&OCID=AIDcmme9zx2qiz_SEM_037eecb058b31669a99b888c6d034e7a:G:s&msclkid=037eecb058b31669a99b888c6d034e7a>