

# **Draw it or Lose it**

# **CS 230 Project Software Design Template**

Version 3.0

## 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 | 09/15/24 | George Saxby | Revised summary, requirements, constraints, and other relevant information necessary for this project. |
| 2.0 | 10/06/24 | George Saxby | Revise and update evaluations (Server side, client side and development tools) |
| 3.0 | 10/21/24 | George Saxby | Update recommendations section of the software design document |

## [Executive Summary](#_sbfa50wo7nsh)

The client aims to transform their existing Android-based game, "Draw It or Lose It," into a multi-platform, web-based game. They want the new version to be built on the foundation of the current Android app and are seeking assistance with environment setup and streamlining the development process. The game must be developed using a cross-platform framework. Additionally, each game and team name must be unique, and only one instance of the game should be allowed in memory at a time. Teams will consist of multiple players, and the game should support either single or multiple teams per session.

## Requirements

The client requires the game to support one or more teams, with each team consisting of multiple players. Only one instance of the game should exist in memory at any given time. To achieve this, unique identifiers must be created for each game instance, team, and player. Additionally, users need the ability to check if a team name is already in use when selecting a name, ensuring that both game and team names are unique.

## [Design Constraints](#_2et92p0)

The primary constraint for this project is developing the game across multiple platforms. Since the team lacks experience with cross-platform development, it will likely require separate teams working on different platform-specific versions based on their expertise. Additionally, the development must meet all client requirements consistently across each platform, ensuring feature parity and proper functionality.

## [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 Entity class serves as the parent (super) class for the Game, Team, and Player classes. As child classes, Game, Team, and Player will inherit the attributes of Entity, while each also has its own specific attributes that are distinct from the parent class.

The GameService class is responsible for ensuring the client’s requirements are fulfilled, such as maintaining only one instance of the game at a time and enforcing unique identifiers (IDs) for the game, team names, and player names.

The ProgramDriver class contains the main method and utilizes the SingletonTester class to ensure the game operates as a singleton (i.e., only one instance can exist at any given time).

The Game class manages a list of teams, while the Team class manages a list of players. The Player class, however, does not manage any lists, as its primary role is to ensure that each player has a unique ID, which can then be assigned to a team. Although players are part of teams, and teams are part of games, the Player class itself does not directly contain references to a team or a game.

**"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** | One major advantage of macOS is its capability to run macOS, Windows, and Linux applications side-by-side, providing flexibility for users working across multiple platforms. Another often-overlooked benefit is macOS's consistency, its interface and system have remained stable over time, making it easy for long-time users to navigate. However, the drawbacks include its higher cost and more limited hardware options compared to Windows or Linux systems, which offer greater flexibility and affordability. | The main advantages of Linux are its wide range of distribution choices and its open-source nature, offering numerous free or affordable options. It is highly versatile, especially for servers and embedded systems, due to its ease of customization. Linux is also known for having superior security protocols compared to Windows or macOS. However, its drawbacks include a limited selection of pre-built machines that come with Linux installed, and occasional file format compatibility issues when interacting with other operating systems. | The key advantage of Windows, especially for corporate users, is its built-in integration with Active Directory-based servers, enabling seamless Windows-based authentication without the need for additional software or costs. However, the drawbacks include limited support for mobile development and weaker security measures compared to other operating systems, making it more susceptible to threats like malware, spyware, and ransomware. | Hosting a web application on a mobile device can be advantageous if the user base is small and the application is simple or lightweight. Android users, in particular, have access to several low-cost web server applications, making mobile devices an affordable option for hosting. However, the drawbacks include the fact that many mobile hosting solutions are cloud-based, which can increase a company’s exposure to security risks and make it more vulnerable to hackers. Additionally, mobile devices as a whole tend to be more susceptible to security threats compared to PCs. |
| **Client Side** | The advantages of macOS include its ease of use once users become familiar with its interface and system. However, a major drawback is its limited availability outside of Apple hardware. Developers who have the skills and training to create macOS applications but don't use Apple devices are restricted, as macOS is not accessible on non-Apple machines, which can limit their ability to work with the platform. | The pros of Linux include its affordability and the high level of control it offers over development. As a free and open-source operating system, Linux is typically easier and quicker to maintain, reducing the time spent on upkeep. However, there are drawbacks to these benefits. Since Linux is open-source and user-managed, security can be a challenge, as users may have to address issues independently, unlike Windows or macOS, which provide dedicated technical support for their products. | The pros of Windows include its wide availability and flexible pricing options, allowing customization based on project requirements. Windows also offers technical support and more advanced security features compared to Linux. However, the cons are that you may need someone with expertise in Windows OS, and additional costs can arise if you require specific features or capabilities. | Although there are numerous applications and tools available for mobile devices, they often fall short in terms of accessibility and the full range of features found on PCs. The advantages include their easy availability at various price points. However, a significant drawback is the wide variety of operating systems for mobile devices, with many applications tailored to specific OS platforms, leading to compatibility issues across different systems. |
| **Development Tools** | macOS utilizes the Swift programming language, with Xcode and Xcode Cloud being the primary tools available for macOS and iOS developers. Xcode Cloud is a service designed specifically for Apple developers, providing teams with a faster and more streamlined way to build, test, and deploy applications efficiently. | Linux offers a vast array of development tools. Given the multitude of options, I will focus on one: Docker. Docker provides a consistent development environment, facilitates the creation of cross-platform applications, and simplifies deployment. Additionally, Docker Hub allows users to bypass the setup of a development environment, enabling them to jump straight into the development process. | Windows was primarily developed in C, with some components written in assembly language. One of the most well-known IDEs for Windows is Visual Studio, which serves as a code editor and, in some cases, a source/version control tool. Visual Studio offers a wide range of tools, and although I was initially slow to recognize their advantages, I now rely on it for development in both my school and work projects. | Java is the primary language for most mobile applications, thanks to its object-oriented features, making it a popular choice among developers. However, Python and C++, along with other variants of C, are also used in mobile development, especially for gaming. While there are many IDEs that support mobile app development, the most popular include VSCode, IntelliJ IDEA, and Eclipse. Personally, I find Eclipse a bit cluttered and prefer using Visual Studio with Xamarin, which enables me to develop cross-platform native applications. |

## 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**: Windows is the recommended operating system for this project, as it integrates effectively with the existing Android application "Draw It or Lose It." Given that Windows accounts for approximately 90% of the global operating system market, it provides access to a broader pool of developers, diverse skill sets, and an extensive range of tools that can facilitate the successful completion of the cross-platform application.
2. **Operating Systems Architectures**: Windows 10 introduced the Universal Windows Platform (UWP), building on the Windows Runtime model. UWP applications can leverage Win32 APIs, the Microsoft .NET Framework, and Windows RT APIs, allowing developers to create a single app that runs across multiple devices using this dual-stack approach. This architecture is recommended for this project due to its versatility and cross-device compatibility.

Reference:

Microsoft. (n.d.). *Examine Windows client architecture*. Microsoft Learn. <https://learn.microsoft.com/en-us/training/modules/explore-windows-architecture/3-examine-windows-client-architecture>

1. **Storage Management**: Server-based storage offers centralized file access, enhanced functionality, and support for failover clustering. It also ensures data redundancy, automates backups, and delivers optimized performance for improved reliability.
2. **Memory Management**: Windows offers various storage and memory management solutions, including Azure Storage. It supports both virtual and physical address space for efficient memory allocation. Additionally, services like OneDrive, Visual Studio, and Azure Cloud can be used for version control and storage management.
3. **Distributed Systems and Networks**: Utilizing a cross-platform development environment can minimize the need for specialized expertise and streamline the application development process. One such option is "Develop 4." To address potential connectivity or outage issues, I recommend ensuring that the servers are built to meet the client's needs, based on forecasts of game usage and user activity once the application is launched in the new environments.
4. **Security**: Given the persistent threat of user data theft, I recommend placing a strong emphasis on security protocols. One effective service for protecting PC, Mac, Android, and iOS devices is Aura. While there is a cost associated with this service, relying solely on the standard security features provided by the operating system is not advisable, as an extra layer of protection is essential for maintaining system safety and optimal performance. Additionally, Aura provides 24/7 customer support based in the U.S., which is a significant advantage of this service.

Reference:  
Microsoft. (n.d.). *What is database security?* Microsoft Azure. <https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-database-security/>