

Win, Lose, or Draw

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template**](#_l6ti7uoag22u)1

[**Table of Contents**](#_30j0zll)2

[**Document Revision History**](#_grjogdjh5fi8)2

[**Executive Summary**](#_sbfa50wo7nsh)3

[**Design Constraints**](#_2et92p0)3

[**System Architecture View**](#_ilbxbyevv6b6)3

[**Domain Model**](#_8h2ehzxfam4o)3

[**Evaluation**](#_2o15spng8stw)3

[**Recommendations**](#_m8aleynsvzvc)5

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 10/11/2021 | Micheal White | Developing web-based version of Draw It or Lose It |

**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 wants to develop a web-based version of Draw It or Lose It that will function across multiple platforms. The app currently exists as an Android only app. As a Technology Consultant, I can support this initiative by tackling the software requirements of The Gaming Room.

## [Design Constraints](#_2et92p0)

1. Web-based applications become more expensive to maintain the more platforms and browsers you develop for.
2. Support and security become more difficult to manage and maintain when developing cross-platform apps.
3. App performance speed can be an issue with web-based apps versus being hosted locally

## [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 is a super class that encapsulates all common traits that are inherited by the child classes (Game, Team, and Player). The ProgramDriver is composed in part by the SingletonTester class which ensures only on instance of a game exists in memory at a time. We can see an association among GameService, Game, Team, and Player to check the unique characteristics in order to ensure only one game instance exists at a time and each team name and player name are unique.

**"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** | Flexibility with terminal commands to configure server or make changes. Strong security. Highest licensing costs. | Similar to Mac but more cost effective. Least amount of apps compared to others. | Highly customizable and widely used. More apps available to use compared to counterparts.  High licensing costs. | Ideal if server is immobile and can be tracked to one place. Cloud services better for scalability but more costly. |
| **Client Side** | Closed system. Moderate expertise and time required. Cost comparable to Windows though slightly higher. | Not widely used. High expertise and time required. Minimum cost being open sourced. | Open system. Minimum expertise and time required. Cost comparable to Mac. | Mobile devices more prevalent than desktops so larger user base. Can be more difficult to implement across many unique devices. |
| **Development Tools** | Eclipse Java EE. Works cross platform with Linux and Windows and with Java ME for mobile devices. One-time costs of $30,000. | Eclipse Java EE. Develop cross platform Windows and Mac and with Java ME for mobile devices. One-time costs of $30,000. | Eclipse Java EE. Works cross platform with Linux and Mac and with Java ME for mobile devices. One-time costs of $30,000. | Current app already built for Android. No JVM running for iPhones/iOS. Use Codename One to compile Java code into native iOS code. Enterprise pricing $399/mo. or $3800 annually. |
|  |  |  |  |  |

**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**: My recommendation would be to use Windows Server on Azure for our “Draw It or Lose It” app. The app is currently available for Android devices and the Windows server platform provides development on Windows and Linux. Thus, using Windows servers would require minimal expertise and cost. There are plenty of IDEs to choose from to develop with as well.
2. **Operating Systems Architectures**: Windows Server comes with a wide selection of tools to help clients manage memory on their devices giving control of how the app is managed on the device. Windows Container provides improved download speeds and performance for larger image sizes. The operating platform supports many different programming languages including Java, .NET, Python, and SQL.
3. **Storage Management**: The Windows server operating system allows users to download the app directly onto their phones memory to allow better performance. The service makes use of cloud services to give our development team plenty of storage for our photo library with the flexibility to add to it in the future.
4. **Memory Management**: The Windows server operating systems provides physical and virtual memory management options. The system allows for between two and four gigabytes of memory on top of what is used to run the application. This allows adequate storage of our 200 8-megabyte photos in physical memory while allowing us to pull an image into the current game through virtual memory to provide a consistent, quality experience for clients.
5. **Distributed Systems and Networks**: Windows App Service is a tool that allows teams to build, deploy, and scale web apps using Java, Python, .NET, PHP and more in containers as well as running Windows and Linux. App Services is tightly integrated with Visual Studio to streamline development. Windows servers operating system offers communication across different workstations and different processors as well as different operating systems. The service also allows developers to deploy updates via slots (staging and production) which redirects traffic seamlessly eliminating downtime. If the changes are not what was expected, then they can be swapped back quickly. Windows offers disaster backup and recovery options and tools for centralized monitoring.
6. **Security**: The Windows server operating system allows for secure management of physical, on-edge, and cloud environments, granting users with account controls to protect data coming in and going out of the system. App Services offers protection including firewalls and application gateways as well as traffic monitoring to auto scale to meet high demand. VPN services also protect clients’ accounts. Windows also has strong anti-spyware software built in to keep unwanted software from infecting the clients’ system.