

<Name-of-Software-Application>

# **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 | 03/17/23 | Robert Golden | Initial Release : Updated all sections |

## [Executive Summary](#_sbfa50wo7nsh)

The client has reached out to us to create a game that is intended to be available on multiple platforms. The game is currently available on Android, but they are asking for it to be web based as well as on IOS. The game is a loosely similar model to the 1980’s gameshow “Win, Lose, or Draw”, and it will be called “Draw It or Lose It”.

## Requirements

* A game will have the ability to have one or more teams involved.
* Each team will need the ability to 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)

The client would like to have the game available on multiple platforms. A list of requirements to be accepted on to the Apple IOS store would need to be considered in the design phase. The customer has a large library of stock drawings that will be displayed, access to this library would have to be designed into the program as well. Since the drawings will be rendered over

## [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 solid black arrow between ProgramDriver and SingletonTester is showing us a relationship between the two, and that ProgramDriver uses SingletonTester. The Entity object above all the other classes that relates to an open arrow is showing us inheritance. The Game, Team, and Player classes inherit the variables of id and name from Entity. Inheriting attributes from the Entity class gives us a way to create unique identifiers thus aiding us in ensuring that only one instance of the game exists in memory at any given time. The black lines between the bottom four classes show Multiplicity relationship, and that each relationship can have 0 to many instances. The fact that there can be 0 to multiple teams related to a game helps fulfill one of the requirements given to us by the client.

**"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** | Despite Mac OS server being discontinued in April 2022 here is my analysis.  + Easy to use command line interface for setting up server configurations.  + High level of security  -The cost can be rather high due to the stringent security requirements.  - more cost could be added with additional servers due to licensing fees  - Use of an Apple Computer is required. | + Free licensing  + Open Source  + more than one option for web hosting  -some users might have trouble navigating the server | + Easy to use and setup  + Most users are familiar with Windows  + Stable  -Expensive for the licensing and for the servers | - lack of resources  - connectivity could be an issue  - initially might be fine, but with growth would experience problems |
| **Client Side** | -could run in to issues if every developer wasn’t using Mac OS  + Apple has good SDK’s but locks you down to the platform and the cost incurred with the platform.  + comes with the security of apple | -while the server costs are a perk it would take a small army of developers to get the project up and running due to the lack of infrastructure that comes with a paid for platform.  + | + this option would the most familiar to any team of developers due to the popularity of windows  -more expensive than Linux  + cheaper than Mac OS | -would require developers experienced in mobile app development  - would require constant updates and would have to meet security requirements from application download platforms. |
| **Development Tools** | Has iCode, and many of the most popular development environments are available.  Since Java is widely available on all platforms it would make sense to chose this option, however there are many other coding languages available on this platform. | Python is the most used language on this platform. A wide variety of development environments including eclipse are available on this platform.  Since Java is widely available on all platforms it would make sense to chose this option, however there are many other coding languages available on this platform. | A wide variety of development environments including eclipse are available on this platform.  Since Java is widely available on all platforms it would make sense to chose this option, however there are many other coding languages available on this platform. | Since Java is widely available on all platforms it would make sense to chose this option, however there are many other coding languages available on this platform.  Would need to use swift and android studio for IDE, and would have to have integration built to communicate with he chosen server platform. |

## 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**: I would recommend using Microsoft Windows as it is a robust operating system that can handle a variety of workloads, and it integrates well with other Microsoft products. Yes, there are some costs incurred with this option, but this option lends itself to easier scaling if the Gaming Room decides to increase the scope of the game yet again.
2. **Operating Systems Architectures**: Windows operating system has many benefits, like a hybrid kernel architecture for instance. This Windows Server architecture provides low-level services to the operating system and applications. The Windows Server kernel provides support for virtualization, networking, security, and other core functions without compromising stability.
3. **Storage Management**: Windows server also integrates well with their azure platform that consists of many storage options including SQL databases, Azure Blob, and quite a few others. Storage can be easily increased if the client intends on scaling.
4. **Memory Management**: Window server uses a variety of memory management techniques to optimize performance and ensure that the client’s software is running smoothly. One of these techniques is virtual memory that allows the operating system to use disk space as an extension of physical memory. Along with the use of Virtual memory, windows also utilize memory paging along with memory compression.
5. **Distributed Systems and Networks**: Since the client would like Draw It or Lose It as a web application as well being available on both major mobile device marketplaces, they will inevitably run in to the issue that phones do run in to connectivity issues. They could handle this by considering how the application will handle connectivity issues such as network outages, high latency, and dropped packets. They could look in to possibly storing some data locally on the device and synchronize it periodically when the network connectivity is restored.
6. **Security**: Overall, windows server provides several tools and capabilities that can help protect user information on and between various platforms. Managing access control is one of these tools that helps the client set user roles and permissions, as well as access authentication. Windows also monitors for security threats via Windows Defender and Windows Firewall which help protect from a multitude different threats. In the unfortunate event that there is a breach in security, utilizing the cloud storage for storing a backup of the system would be crucial in minimizing the downtime of the system.