

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 | 01/25/2024 | Taya Zaragoza | Addition to summaries and project overview |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room wants to adapt their Android-only app, Draw It or Lose It, into a web-based game that can be played across multiple platforms. The game is ran in four rounds, each lasting for a minute long. In each round, the game will choose a word or phrase and render pictures from a large library to give as clues. The images are rendered at a steady rate and are completed in 30-seconds. If the playing team does not guess the correct word/phrase after the minute, then the opposing team will be allowed one guess from each player, with a 15-second time limit.

## Requirements

* Game must have one or more teams involved
* Each team will have multiple players assigned
* Game and team names must be unique
* Only one instance of the game can exist
* Must be available on multiple platforms

## [Design Constraints](#_2et92p0)

Some design constraints to consider are how the game needs to be available on multiple platforms. This means that all components used must be compatible with all operating systems and able to run together at the same time. This limits the amount of features that can be used overall. Because the applicant is now web-based, all users must be connected to the internet in order for the game to work. This also means that servers for the game must be able to handle the traffic.

## [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)

In this UML diagram, we can first see that the Program driver uses the singleton tester. The GameService class, Game class, Team class, and Player class all inherit from the Entity class. The GameService class to the Game class has a zero to many relationship. As does Game class to Team, and Team to Player.

**"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** | Offers better security and reliability | Open-source nature, less targeted | Requires regular updates which can lead to temporary downtime | Offer faster speeds, requires more maintenance |
| **Client Side** | A good middle ground in expense | Cost-effective, customizable, requires lower maintenance | Usually more pricey due to licensing and updates | Generally most expensive, profits must be shared with the app store the app is listed on |
| **Development Tools** | MacOS can support many different programming languages that Linux and Windows may lack. Mac is known to be smoother and more aesthetic for coders. | Has many good IDEs such as Emacs, Eclipse, and Code::Blocks. More command line tools, scripting capabilities, and you can use your terminal parallel to your IDE | A popular IDE for Windows is Visual Studio or Mono/Mono Develop. C#, Java, and C++ are common programming languages. | Popular programming languages are Java, Python, and C++ for Andriod. Popular languages for iOS are Swift, Objective-C, and HTML5. Both operating systems can use C#. |

## 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**: Linux is the recommended operating platform due to it’s low-cost and customizability.
2. **Operating Systems Architectures**: Linux has four fundamental layers in its architecture. It has the application, shell, kernel, and hardware.
3. **Storage Management**: An appropriate storage management for Linux would be xfs.
4. **Memory Management**: Linux will be using memory management to track points, rounds, and game and team names.
5. **Distributed Systems and Networks**: Linux uses the TCP/IP suite from Unix that allows users from multiple platforms to access the application and share memory. All other operating systems follow this protocol so that the web can be accessed from all users.
6. **Security**: Linux ensures security by assigning lower privileges to users, tracks activity and access, and remains secure with its open-source coding. Viruses have a harder time getting system-level access.