

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 | 07/23/2024 | Elise Parramoure | Updated the software design document regarding executive summary, requirements, domain model and design constraints |

**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)

Draw It or Lose It is a game in which teams of multiple players compete to guess what image is being drawn. The software will pick an image from the library and slowly render the drawing at which time, teams will take turns to guess the image before the 30 second timer runs out. If the team does not guess correctly by the expiration of the timer, the opposite team will get one guess within a 15 second limit.

## Requirements

*1.The game will be able to accommodate one or more teams.*

*2. Each team will have multiple players assigned to it.*

*3. A list must be created to contain all the team and game names.*

*4. Game and team names must be unique so that users can check whether a name is on the list and in use when choosing a team name.*

*5.Unique identifiers must be created for each instance of the game, team, or player to ensure only one instance of the game exists in memory at any given time.*

## [Design Constraints](#_2et92p0)

The image library can not be unlimited so the number of images included will be constrained.

There are timers in the game so we will have to have the software operate within time constraints.

The staff at the Gaming Room creates a design constraint as they do not know how to set up the environment.

## [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 beginning class is Entity and Game, Team, and Player all inherit traits from this parent class. The class includes an ID and a name as attributes, along with methods for getting the ID, getting the name and writing to a string, in addition to the method Entity. The Player class has no other attributes beyond the inherited ones. However the team class includes the Entity attributes as well as a list of Players it attains from its relationship to the Player class. In a similar fashion, the Game class inherits from Entity and contains the attribute for a list of team names that it attains from Team class. The GameService class does not inherit any attributes from the aforementioned class, yet it does have a relationship with the Game class where it attains a list of games. The GameService attributes include the GameId, PlayerID, TeamID, and service as well.

**"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** | Able to run all three operating platforms side by side. Always quality and security is high level. | Open source with many free and cheap options, however few machines have this system-built in. | Quick to collect malware, spyware, and bugs. However it supports more hardware and software than any OS. | Slower operating speed |
| **Client Side** | Most expensive OS of the three generally with less frequent updates.Ties in well with Apple devices and mobile phones. | Linux is free so anyone can download and use it cutting costs. | Cost effective, cheaper than Mac systems, with many hardware options. Also installed on most computers and laptops. | Expensive and often requiring updates to keep up with the speed. |
| **Development Tools** | Document based opposed to program based so every app has the same menu. | Essentially runs all and any needed apps easily but more time is required to set Linux as your primary operating system. Also can be an issue that Linux runs with a Chrome browser. | Built well to help develop games and applications. Prized for its convenience and ease of use and customizability. | Easily transported and more often used on the go |

## 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 Windows Operating System for Draw It or Lose It. Windows OS offers a great variety of features and performance options. For example, you can mute your notifications while in a game. Another awesome and useful feature of the Microsoft Windows OS DirectX. This is application programming interfaces or APIs that allow for access to memory, video cards, sound cards and more for fine-tuning and customization for gamers individual performance. It is also the most popular of the platforms meaning more people are familiar with the operating system.
2. **Operating Systems Architectures**: This operates on a microkernel architecture which separates functions into components with specific functions. This provides the system with more stability and security. Most operating systems actually seem to operate with the same architecture type, or at least very similar, however, the added stability from microkernel architecture is appreciated inn a game with many players on multiple platforms.
3. **Storage Management**: Since the game application is web-based, I would recommend Cloud storage. This allows for access from multiple devices and utilize only the storage space needed to keep the game running during the instance we are in. If the game becomes increasingly popular and we need to expand our storage, Cloud storage allows for this simply.
4. **Memory Management**: Memory management controls the movement between paging files and physical memory and deletes the physical file in the space once the game ends. I would recommend Java for our main programming language as it automatically manages memory with the garbage collector.
5. **Distributed Systems and Networks**: Windows is built well to adapt to games and sharing to multiple networks. With the cloud-like system, we can use it to handle jobs dealing with multiple CPUs as would be used in this web-based game application.
6. **Security**: The Microsoft PKI (public key infrastructure) verifies the identity of users, while the firewalls and built-in antivirus blocks unwanted connections, ads, and spyware. The purpose of the PKI is to encourage and facilitate a secure transfer of information within a network range. This will protect our code as well as the users account information.