

# 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 | 07/21/2024 | Stephanie French | Software Design Document   * Executive Summary * Design Constraints * Domain Model |
| 1.1 | 08/04/2024 | Stephanie French | Software Design Document   * Development Requirements |
| 1.2 | 08/18/2024 | Stephanie French | Software Design Document   * Recommendations |

## [Executive Summary](#_sbfa50wo7nsh)

<Write a summary to introduce the software design problem and present a solution. Be sure to provide the client with any critical information they must know in order to proceed with the process you are proposing.>

The Gaming Room has a game, Draw It or Lose It, only available to Android users. Creative Technology Solutions has been enlisted to develop a similar game that is web-based and able to serve multiple platforms.

## Requirements

*<* Please note: While this section is not being assessed, it will support your outline of the design constraints below. *In your summary, identify each of the client’s business and technical requirements in a clear and concise manner.>*

## [Design Constraints](#_2et92p0)

<Identify the design constraints for developing the game application in a web-based distributed environment and explain the implications of the design constraints on application development.>

Web-based

Works on multiple platforms

Four rounds of play, lasting one minute each

Ability to gradually render images from a library of drawings, completing and expiring at 30 second mark

Allow remaining teams 15 seconds to offer a guess if current team cannot complete puzzle

Support one or more teams

Ability to assign players to teams

Require Teams/Players to select a unique name

Check that each Team/Player name is unique

Check that only one instance of the game exists in memory at any given time

## [System Architecture View](file:///C:\Users\steph\Downloads\CS%20230%20Project%20Software%20Design%20Template%20(3).docx#_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)

<Describe the UML class diagram provided below. Explain how the classes relate to each other. Identify any object-oriented programming principles that are demonstrated in the diagram and how they are used to fulfill the software requirements efficiently.>

* ProgramDriver contains the main method.
* SingletonTester ensures that only ONE instance of the game exists.
* There are multiple principles/concepts of object-oriented programming used in this diagram.
  + Inheritance is represented in the relationship between Entity, Game, Team and Player. The Entity class serves as a parent class. The Game, Team and Player classes serve as child classes. The child classes inherit all fields and methods of the parent class but they can also implement their own as well.
  + Aggregation is represented in the relationships between GameService, Game, Team and Player (if an object from one these classes is deleted it will not interfere with other objects in other classes)
    - GameService has 0 to many games
    - Game has 0 to many teams
    - Team has 0 to many players

**"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** | \*\*Has server configurability that is easy to access/use.  \*\*Stronger security protocols than its competitors.  \*\*Number of users vastly smaller than counterparts.  \*\*Most costly than other options. | \*\*Affordable  \*\*Free and open source.  \*\*Some applications may not be supported. | \*\*Strong support resources.  \*\*Easy to set up  \*\*Closed source | \*\*Portable  \*\*Not as well equipped to support a server |
| **Client Side** | \*\*Great integration with other apple products.  \*\*Is not compatible with products that are not apple.  \*\*Again, is more expensive. | \*\*Affordable  \*\*Quick loading  \*\*But requires more skill to navigate. | \*\*User friendly  \*\*More vulnerability to viruses | \*\*Vastly different interface for users.  \*\*Portable |
| **Development Tools** | \*\*Many programming languages/IDEs are available | \*\*Many programming languages/IDEs are available | \*\*Many programming languages/IDEs are available | \*\*Framework for many languages have been built |

## 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 an appropriate operating platform that will allow The Gaming Room to expand Draw It or Lose It to other computing environments. There are many IDEs available, it is user friendly and has great support.
2. **Operating Systems Architectures**: The OS is divided into two parts, general user and then kernel. General user is what the user sees and interacts with during their day-to-day work. Kernel is on the backside of the OS and deals with management of hardware and memory as well as inputs/outputs.
3. **Storage Management**: Cloud based storage is recommended. The cost of a cloud-based system is more efficient in the long run because with physical storage you need to update servers and drivers or repurchase servers once they reach their lifetime.
4. **Memory Management**: Implementation of virtual memory will allow temporary movement of data that is used less frequently from RAM to disk. By freeing up RAM you are increasing speed and performance of the game.
5. **Distributed Systems and Networks**: Unity is a game engine that was released in 2005. It can support MANY platform types and is user friendly. Their website states, “Create once, ship anywhere.” You can build your game with this engine and deploy it across AR, VR, mobile, desktop and console platforms. This engine is great to use now AND if you would like to expand the company to platforms outside of mobile/desktop there are other options.
6. **Security**: Windows has its own built-in security software but there are other options like McAfee or Norton.