

Draw it or Lose it

# **CS 230 Project Software Design Template**

Version 1.1

## 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 | 11/18/2023 | Robert Bell | Created Document |
| 1.1 | 12/02/2023 | Robert Bell | Updated Logo |

**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 application similar to the 80’s game show Win, Lose or Draw. The app will render images from a stock library over the course of 30 seconds. Points awarded to the team which guesses the image first. If no team guesses correctly before the image is rendered completely, remaining teams have 15 seconds to make one guess. The game will be played over four rounds.

## Requirements

The application needs to be web-based.  
The game needs to be able to add one or more teams involved in the game.  
Each team needs to allow for multiple players to be assigned to it.  
Game and teams names need to be unique and alert players if a name is already in use.

Only one instance of the game can exist in memory at a time.

## [Design Constraints](#_2et92p0)

Hardware requirements will be determined during development. The web-based version of the game will need to be functional within most common web browsers and ideally should work on mobile browsers as well. Some browsers may have unique requirements and thus the application may need to detect the user’s browser and adjust for compatibility. The server will need to store team names and cross reference new names against them. Each game will need to be accessible to multiple players playing on multiple devices simultaneously, meaning most of the game will likely have to be run server side.

## [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)

Each of the classes is in a “has a” aggregation relationship with the class to its left. GameService can have multiple Games, each Game can have multiple Teams, each Team can have multiple Players. Though as shown in this diagram a Player doesn’t need to be part of a Team, a Team doesn’t need to be part of a Game, and a Game doesn’t need to be part of a GameService. Player, Team, and Game all use the Entity class.

**"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** | Uncommon, based on UNIX, requires licensing. May be hard to find people qualified to develop for it. | Linux has become the go to platform for hosting servers. Low overhead, requires expertise but due to its popularity in server applications that expertise should be relatively common. | Probably the second most commonly used for server applications. Requires licensing. | The only potential advantage would be if we want users to be able to host their own servers. Otherwise hosting on mobile devices would be very costly for no benefit. |
| **Client Side** | Due to being web based, individual platforms should require minimal porting for different client operating systems. Though the Safari browser will need to be supported. | Support for Chrome, Chromium, and Firefox web browsers will need to be implemented for most Linux distros. These browsers are also commonly used on other platforms. | Support for Microsoft Edge browsers will need to be implemented for Windows clients. | Interface will need to be designed to be readable and functional on both desktop and mobile. Considerations should be made for input limitations inherent to mobile devices as well as any limitations of mobile web browsers. |
| **Development Tools** | Some combination of CSS, HTML 5, PHP, and JavaScript will need to be used for all platforms.  PyCharm, Visual Studio Code | Some combination of CSS, HTML 5, PHP, and JavaScript will need to be used for all platforms.  Visual Studio Code, PyCharm | Some combination of CSS, HTML 5, PHP, and JavaScript will need to be used for all platforms.  Notepad++, Visual Studio Code, PyCharm | Some combination of CSS, HTML 5, PHP, and JavaScript will need to be used for all platforms.  Most likely will not be developing on mobile only for mobile. |

## 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**: The server should be written for and hosted on Linux while being accessible by clients on every major operating system via their web browser.
2. **Operating Systems Architectures**: Given this will be a web-based application the architecture for the server side application will be fairly similar regardless of OS.
3. **Storage Management**: LVM (Logical Volume Management)
4. **Memory Management**: Memory sharing can be implemented in Linux to allow multiple processes to access the same portion of memory, this may be necessary to allow the server to draw from the same photo library while running multiple games.
5. **Distributed Systems and Networks**: The overwhelming majority of the application will be server side with all clients dependent on a constant connection to the server. Considerations need to be made for temporary interruptions in connectivity, allowing clients to reconnect, and allowing games to continue when clients have dropped or ending those games if all clients drop.
6. **Security**: The first component of user protection is not collecting unnecessary information from the user. All user information ought to be inaccessible to other clients. Linux has built in permissions considerations for different users connecting to the server, though we would likely use a different solution.