

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 |
| --- | --- | --- | --- |
| 1.1 | 2/10/23 | Steve Smith | Summary and Design Constrants |
| 1.2 | 02/18/23 | Steve Smith | Development Requirements |
| 1.3 | 02/24/23 | Steve Smith | Recommendations |

**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 has decided to develop a web-based game. At the moment the client’s game is only available on an android application. The application they are wanted to make is currently called: Draw it, or Lose It. The client is hoping to serve all platforms and not just android. The idea for the game was based off a 1980s television game: Win, Lose, or Draw.

## 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)

The Gaming Room design constraints:

* Players draw images on an easel to help team member guess the puzzle (phrase, title, or thing)
* The game consists of four rounds lasting one minute each.
* Drawings are completed at the 30 second mark.
* If the team does not guess the puzzle before the time expires, the remaining teams will have one guess 15 seconds to solve
* The client requires the app to be web based.
* A game will have the ability to have one or more teams involved.
* Each team will have multiple players assigned to it.
* Game and team names must be unique.
* Only one instance of the game can exist in memory at any given time.
* Create unique identifiers for each instance of game, team, and player

## [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 Entity class is the parent class to Game Team and Player classes. The classes inherit characteristics from the Entity class. The ProgramDriver class drives all the classes with the main() method.

**"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** | The pros are flexible terminal to configure the server. Easy to use interface. The cons are high in cost and a closed system with no hardware updates. | The pros are flexible terminal to configure the server. Low in cost. More secure than windows. The cons are learning curve is overwhelming. Compatible difficulties. | The pros are good page load time, predictable processing performance, and fewer browser compatibly issues. | The advantages are high speed and performance, offline access availability, and real time notifications. The disadvantages are cost of development and maintenance. Can take up storage. |
| **Client Side** | High in cost. Difficulty for users to learn with needed skills. | No client-side precautions need to be made. Minimum cost. Difficulty for user to learn with needed skills. | Cost is more expensive. User friendly and easy to understand. | The advantages are easy of use and convenience. Low cost for the client. The disadvantage is limited function. |
| **Development Tools** | The frontend languages are HTML, CSS, and JavaScript. Java runtime development kit will be needed. Java IDE will be needed. | The frontend languages are HTML, CSS, and JavaScript. Java runtime development kit will be needed. Java IDE will be needed. | The frontend languages are HTML, CSS, and JavaScript. Java runtime development kit will be needed. Java IDE will be needed. | The frontend languages are HTML, CSS, and JavaScript. Java runtime development kit will be needed. Java IDE will be needed. |

## 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 the Linux Operating Platform for the Gaming Room’s “Draw it or Lose it” application. The operating system is free and open source. Linux is more secure with no precautions needed by the user.
2. **Operating Systems Architectures**: The Linux operating system uses Extended File System and Unified File System. The system is open sourced which provides more security for applications. The architecture is composed of elements such as the Kernel, System Library, Hardware layer, System, and Shell functions. There are a set of system library functions. These libraries are implemented by the operating system and do not require code access rights on the kernel modules.
3. **Storage Management**: Linux uses Network File System (NFS) to manage storage. This enables system administrators to consolidate resources onto centralized servers.
4. **Memory Management**: The Linux memory management uses virtual memory, memory allocation for kernel internal structures and user programs, and mapping of files into processes address space. The virtual memory eliminates physical memory size constraints. This allows apps to load and run faster.
5. **Distributed Systems and Networks**: The Linux operating system uses Kubernetes to create server nodes the game application can run on. Kubernetes is an open-source system for automating deployment, scaling, and management of containerized application. This can reduce the risk of system outages. Load balancing spread clients requests evenly across app servers which imptr
6. **Security**: The Linux operating system has different set of tools for security. They include network tunneling, sniffing, scanning and mapping.