

DRAW IT OR LOSE IT

# **CS 230 Project Software Design Template**

Version 2.0

## 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 |  |
| --- | --- | --- | --- |
| 1.0 | <05/25/2025> | Rudolph Travers |
| 2.0 | <06/07/2025> | Rudolph Travers |
| 3.0 | <06/20/2025> | Rudolph Travers |

**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 a strong desire to broaden its user base by developing their current game *Draw it or Lose it* into a web-based game that operates on multiple platforms. Currently*, Draw it or Lose it* is only available on android. The plan is to implement a library of stock images that the user will use to play the game. The Game room knows that this will make the game faster paced and improve the user experience.

## Requirements

* 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 to allow users to check whether a name is in use when choosing a team name.
* Only one instance of the game can exist in memory at any given time. This can be accomplished by creating unique identifiers for each instance of a game, team, or player.

## [Design Constraints](#_2et92p0)

The gameplay of the Draw it or Lose it is straightforward, there are four, one-minute rounds. The photo is rendered at a steady rate but fully complete at the 30 sec mark of the round. If the team can’t guess the photo in the minute. The opposing team have fifteen seconds to give a single guess.

## [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 UML diagram gives a breakdown of how “gaming rooms” work. It gives insight into each of the classes and their attributes, the methods, and how the classes work with each other. The + and – show you what is public or private. Game, Team, and Player are all classes that inherit attributes from Entity.

**"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** | **Strengths** - *User-friendly interface that is aesthetically pleasing.*  *Simple UI that provides ease of use for users*  **Weaknesses** *– not ideal for web hosting services* | **Strengths** – *cost effective, robust security features,*  **Weaknesses** *– not as much support for developers as other Oss, not ideal for web hosting services* | **Strengths** – *wide array of available software/High user accessibility*  **Weaknesses** *– more susceptible to virus and malware* | **Strengths** – *popularity among users and cost effective*  **Weaknesses** *– weak security/high array of different products* |
| **Client Side** | **Strengths** – *requires specialized knowledge and developers/moderate cost* | **Strengths** - *low cost*  *Weaknesses - require developers with specialized knowledge of the OS* | **Strengths** – *no specialized expertise required of developers.* | **Strengths** - *requires specialized knowledge/minimal cost* |
| **Development Tools** | Languages   * Java * C++ * Python   IDE – pycharm/vscode/eclipse | Languages   * Java * C++ * python | Languages   * Java * C++ * Python   IDE – pycharm/vscode/eclipse | Languages   * Java * C++ * python |

## 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 optimal operating platform for hosting the game is the Linux platform. Because Linux is an open-source platform there is an extensive community of knowledgeable developers. Linux is known to be the best OS for server applications; this is since many distributions are free to use, security is increased because source code is constantly monitored by a community of developers. and its servers are scalable. All of this results in Linux being a cost-effective choice and is the best for this type of application.
2. **Operating Systems Architectures**: At its core, Linux is built around a monolithickernel. This allows for direct interaction between hardware and resources providing a high degree of efficiency. The Linux kernel is the engine of the operating system; it manages CPU scheduling and memory which allows the OS to run multiple applications simultaneously without any lag.

1. **Storage Management**: Since we are building Draw it or Lose it as a web-based application the ideal solution for storage management is cloud storage. This is a great solution because it allows us to scale storage based on demand, if storage needs increase it can be expanded with zero effort. Google cloud is an option that fits all our needs.
2. **Memory Management**: Java will be the primary language for the backend development of the application. Java will use the garbage collector to manage the memory system.
3. **Distributed Systems and Networks**: In order to reach users on any device with internet access the application will run on a web server that can be accessed with any web browser. A backup server will also be implemented to enhance reliability and create redundancy.
4. **Security**: To make sure that our application provides a high level of security for our users HTTPS will be implemented for secure data transmission between clients and servers. Like previously stated, the Linux OS has an expert community that is reviewing source code continuously so any security risk can be found and dealt with rapidly.