

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template**](#_l6ti7uoag22u)1

[**Table of Contents**](#_30j0zll)2

[**Document Revision History**](#_grjogdjh5fi8)2

[**Executive Summary**](#_sbfa50wo7nsh)3

[**Design Constraints**](#_2et92p0)3

[**System Architecture View**](#_ilbxbyevv6b6)3

[**Domain Model**](#_8h2ehzxfam4o)3

[**Evaluation**](#_2o15spng8stw)3

[**Recommendations**](#_m8aleynsvzvc)5

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 11/08/2022 | Stephen Cessna | Updated the Template for the new client “The Gaming Room” and their game “Draw It or Lose It” |

**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)

This company wants to develop a web-based game that can be played on multiple platforms. As of right now the game is only available on Android. This game is called “Draw it or Lose it” it also consists of multiple teams with each game having four rounds lasting one minute per round. A picture is pulled from a library of images where one team will guess the image until the time runes out. If that team does not manage to guess the image before the time runs out, then the other teams will have 15 seconds to offer one guess to sole the puzzle.

## [Design Constraints](#_2et92p0)

One design constraint is that each team will need multiple players. Another design constraint would be that each team will have to come up with a unique team name that is not already in use. Another constraint would be to get this game compatible across all platforms. Another constraint to think about would be that you would need at least two teams for the game to start.

## [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 class shares a common reference like the name and id of the class, making the class Entity the superclass. The class Entity creates a bond between Game, Team, and Player classes which means all those classes will inherit information from 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** | Flexible terminal commands to configure the server, access, or make changes to the server. Has a moderate learning curve. | It is also flexible with its terminal commands with an upside of being more cost friendly. However, it has the hardest learning curve. | Offers more software’s compared to the other OS mentioned. | It would be better for the server to not be a mobile device since it has poor security and accessibility compared to the other three options. |
| **Client Side** | The cost would be similar to Windows, but it is moderately easy to use, and the time spent would be minimal as well. I think that mac would work well just not quite as good as Windows. | The cost for Linux wouldn’t be too much different from Windows and Mac however the time and expertise that are necessary is far more than Windows or Mac do use. Linux has a different learning curve than the others. | The cost for Windows would be about the same as Mac however I personally feel that Windows has more features and isn’t too hard to learn. | This would provide more flexibility to the clients and developers in the regard that you can access this from anywhere that you have a cell signal. |
| **Development Tools** | The most common programming language for writing applications on Mac is Swift with the IDE Xcode or Atom | The most common programming language for writing applications on Linux is C with the IDE Eclipse or Atom | The most common programming language for writing applications on Windows is C++ as well as HTML and CSS with the IDE Eclipse or Visual Studio | Then depending on what device, you are using on mobile will determine the most common language so for IOS the most common language is Swift and then for Android the most common language is Java and Kotlin with the IDE Visual Stuido |

## 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**: My recommendation for the project would be to start on Windows. The reason for this is because Windows has a lot of features compared to all the other platforms and has a lot more availability when it comes to languages and IDEs
2. **Operating Systems Architectures**: Windows provides great services that can be used by everyone. It has many applications that show Graphical User Interface while still accessing system resources.
3. **Storage Management**: Windows has a feature called storage sense that allows you to manage your files on the hard drive as well as windows offers a cloud storage system if needed.
4. **Memory Management**: When creating this application you will need to create a database or a library for the pictures allowing you to keep the pictures and the program on the same drive.
5. **Distributed Systems and Networks**: Once the software for the game has been created you can upload it to the web servers allowing the other platforms to connect to that webserver which would allow cross-play. The company would most likely have to do a stress test to make sure that their server can handle the number of clients that would be using their software and they may have to investigate upgrading or purchasing more servers depending on the results of the test.
6. **Security**: Windows comes standard with a nice built-in security protection software. However, to secure your user data I would recommend looking for a better alternative. Windows security software is great when scanning for malware, viruses, and security threats.