

<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 | <09/28/23> | <Rowan Lundquist> | <Draw it or lose it plan > |

**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 created an idea for a game called “Draw It or Lose It. The game will have 1 or multiple teams of two or more people. The game will render images from a large library for the teams to guess a puzzle which will be a phrase, title, or thing. There are 4 rounds lasting 1 minute each. If the team does not guess the puzzle the other teams will have 15 seconds to steal and guess the puzzle, but they only get one guess. Draw It or Lose It is only available on android for the time being. The Gaming Room wants us to develop the environment and game.

## Requirements

* The 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)

Developing this game application in a web-based distribution environment will be challenging because we will need to completely overhaul the code for the game. If we were trying to port the game from android to apple there are many applications, we could use to do this but since we are creating the game to be web based we will have to redo most of it. We also need to create the server and environment which will create an extra barrier before we can release the product to our client. We will also need to create unique identifiers for each instance of a game, team, and player so that only one instance of the game exists in memory. This can cause extra issues and problems while developing the game but should not be a huge problem. We will face some challenges, but I feel we have the resources, and our team has the skills to achieve this within the time frame our client expects!

## [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)

<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.>

The Entity class is the parent class of the Player, Team, and Game classes. This means they inherit all variables and functions without having to incorporate them into their own class. Game service is and independent class and has most of the functions and variables. GameService is associated with Game, Game is associated with Team, and Team is associated with Player. All of these associations also show that Game, Team, and Player can all have 0 cases or as many as necessary. Finally we have the ProgramDriver class has a main function and has a fully filled arrow pointing at the SingletonTester class which means that ProgramDriver uses SingletonTester but SingletonTester does not need ProgramDriver.

**"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 terminal commands will improve the ease of creating the server and making changes to the server. Mac is easily upgradable and has a multitude of different server hosting needs. A disadvantage could be that other web hosting services are more widely used. Mac will require a licensing fee which will increase the cost for the client. | Linux shares many characteristics with Mac, but it is much less expensive. A huge advantage of Linux is the security, and this makes it the most popular server hosting service. A disadvantage is that Linux is the most challenging OS to learn and operate. Linux servers can often be cheap due to distributions generally being free while other operating systems require a licensing fee. | Windows has been the most used OS for a long time which means many more people excel in it. Windows can often outperform other operating systems in terms of loading times and resource management. The biggest disadvantage would be the security as it is easy to get a virus on this OS. Windows will require a fee for each server so this will increase the cost for the client. | Mobile devices have many technical restrictions compared to the other Operating Systems. The advantages are the low price and how easy it is to setup and maintain the server. The main characteristic of a mobile device is that the server would be mobile and have a greater outreach due to that. The servers for mobile devices will depend on many factors but can be from just above $50 to about $350 a month. This will have to be a decision the client makes depending on what their vision is. |
| **Client Side** | Mac can be more expensive than windows for the same quality. Need an expert to make sure that the program is compatible with Mac and the other Operating Systems. Offers different software that can often be better than what is available on other Operating Systems. | Linux can be the most complex OS. Linux also has many different distributions which can lead to confusion and problems if not educated on the OS. Cheapest of all operating systems and very widely used in the industry. Will need expertise to maintain, but since it is very widely used most developers will be able to work with Linux. | Windows is seen as the simplest OS and is very comparable to the others. The price is similar to Mac but can vary based on different variables. Most developers are familiar with programming with Windows which can lead to less problems through the development process. Will have | Will need a team who knows how to develop that game on mobile operating systems, but not as hard to learn as other OS. The cost will could be lower than apple and windows. Could require less maintenance if launched without game breaking bugs and if no huge updates are needed. |
| **Development Tools** | Java would be the best language to use for this game. We could use many different IDEs, but I would recommend eclipse. We are also going to need to use HTML to make the website to host the game. | Java would be the best language to use for this game. I would still use eclipse with Linux, but the different distributions could vary. Will need to use HTML to create the website to host the game | Java would be the best language to use for this game. I recommend the use of eclipse on Windows. Will need to use HTML to create the website to host the game | Java would be the best language to use for this game. I recommend using eclipse for this project. We will need to port the game to IOS through a application development framework. |

## 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 recommend using Windows for this project. Windows will be the most valuable OS for our team and client because of the ease of use and the adaptability of the OS. It will cost money to maintain the server, but this is well worth to have a server the client can rely on.
2. **Operating Systems Architectures**: Windows has many features and services that will help our team create this application. Windows GUI is very simple and will allow the team to see what we are working on. We can also use Windows built-in security measures to help our client stay secure. Windows can be accessed with many different user accounts, which will help with adaptability since people will not always be using the same computer.
3. **Storage Management**: Windows built in Storage Management is top of the line in the industry. We can manage our files through different drives, different folders, and we can search for what we need whenever we need it. If this is not enough for our client there are also many storage management applications that are free and available on Windows to use although I feel as though we would not need them. We can also use the cloud to store files, which will be very important to keep everything organized throughout the company and team. Cloud storage will also make it an easy process for teams to share information and documents with others.
4. **Memory Management**: The Ram of the system will handle most of the Memory Management, so we need to make sure each system has adequate Ram to do so. This will be a very important step in the process because of the library of images that are going to need to be stored and accessed often.
5. **Distributed Systems and Networks**: Since this game is already available on Android it will be very easy to port to IOS using the “app store” although we may need to run some tests and change some code. It will be an easy process to create the website using a language like html or JavaScript and adding our game to that website. This will make it so we can run the game on any OS. If the client decides they want to make the game downloadable we could put it on stores such as “steam” so that any OS could run it if the pc had hardware that matched what is required.
6. **Security**: Although Windows is not the most secure OS it still has many high-level security features that are built into the OS. Another step that we can take to make sure everybody’s information is secure is to either not collect user information or not save that information to our server but instead delete it once the user logs off. We can also encrypt user information and have that information accessible through a password.

Works Cited

* Lundquist, R. (2023). “CS 230 Project one Software Design [Unpublished document]. Southern New Hampshire University.