

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 | 11/08/2021 | Denise Kendall | The Gaming Room would like to convert their gaming app, Draw It or Lose It, into a web-based game. |

**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 gaming app on Android named Draw It or Lose It, which has a large library of stock drawings. They would like to transfer it to a web-based game that can run on multiple platforms. The game needs to support multiple teams as well as players. Each game and team names need to be different and only one game is allowed.

## [Design Constraints](#_2et92p0)

* Programming Language
  + For this game, we need to use a programming language can create a cross-platform software. This way, we can stay organized using only one language rather than multiple languages on different platforms. We need to choose this carefully since we must consider if and how many players must download extra software to be able to read and run the game based on the language we chose.
* Operating System
  + The game must be able to meet the standards of each platform since the client wants to be available to multiple platforms.
* Cloud Environment
  + This game has a large library of stock drawings that are clues for the players. A decision must be made on how it is stored. We need to have enough room to be able to store all the drawings. We need to know if it will be through an outside source and if there are limitations or cost.

## [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 ProgramDriver is the main class that will hold all the instructions for the game. It will use the SingletonTest class so that we can only create one game. The Entity class is the parent class to Game, Team, and Player classes. They will inherit all of the functions of the Entity class so that you don’t have to retype the same code or you can have a basic code and have each child class make change it to their specific need. The GameService class is associated with the Game class. The GameService class will hold the setters and getters for the game information. The Game, Team, and Player classes are all associated with each other and have a zero to many relationship.

**"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** | An advantage is it has open source apps, but it doesn’t have very good cross-development support. | Linux is an open source operating system and secure, but it isn’t as developed unlike Windows. | Windows is consistent and flexible, but we could run into compatibility issues. | One advantage of using mobile devices is that it is mobile. However, they are limited by their battery life and have a slower processor. |
| **Client Side** | Depending on how fancy the developers want to get it could take anywhere from 2 weeks to a couple of years to create a game. | I think one of the things to consider, if Linux is chosen, is finding people who know how to use Linux. It is the least known among Mac and Windows. | Based on previous games, the cost to creating a web-based game can cost several hundred thousand dollars. | The estimated time to create an app for a mobile device is 9+ months and cost anywhere from $10,000 to $500,000. |
| **Development Tools** | Visual Studio Code is a great tool for Macs because it covers several different languages. Lua would be a beginner-friendly language to learn and it is support by Mac. | C++ is flexible and has great memory management. Eclipse seems to be the popular choice to use as an IDE for Linux. Eclipse has great debugging capabilities. It can also handle big projects. | C# is a popular programming language for gaming engines. Eclipse is a free IDE that can be used for Windows, Mac and Linux. It is customizable which can be helpful. | JavaScript can be used to create mobile apps, including iPhones and Android. Xamarin is a cross-platform mobile app development, however it does have a monthly fee. |

## 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 using Linux as the operating system. It is a lower cost and it has great security.
2. **Operating Systems Architectures**: The Linux Operating System’s architecture consists of the kernel, hardware layer, system library, shell and system utility. The kernel takes care of the major activities of the operating system. The hardware layer is made of the physical parts of the computer. The shell is used to communicate between the user and the kernel.
3. **Storage Management**: I would use a cloud storage software. You will have more space to work with and it can’t be easily destroyed. We have options of either open source or we pay a fee depending on how much space we need.
4. **Memory Management**: Each process has its separate virtual address space which is directly linked using memory mapping. Linux also uses large address spaces and lets each running process in the system a share of the physical memory of the system.
5. **Distributed Systems and Networks**: We can use a software such as Pop!\_OS to distribute the game. Pop!\_OS supports both AMD and Nvidia GPUs. It has built-in GPU support. I would also suggest using a cloud network for consistent connectivity and security.
6. **Security**: One reason why Linux is secure is because it is open source. Anyone can look at it and check for holes in the system. This also means that it is consistently updated and free. When downloading or opening files, Linux will give the user a warning if it looks suspicious and will need permission to continue.