

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 | 09/18/2022 | Tanya McHenry | Updated the requirements, executive summary, and design constraints |
| 1.0 | 10/2/2022 | Tanya McHenry | Updated Evaluation |
| 1.0 | 6/15/2023 | Tanya McHenry | `Updated 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 is implementing a web-based version of the game Draw It or Lose It, which is currently on an Android app only. The game will allow multiple players, but only one user in the memory at a time. This document provides information on the operating constraints in the product delivery.

## [Design Constraints](#_2et92p0)

Multi-Platform Design and Distribution – With the game being set up on a web-based version, we will need to include ways for different computer operating systems to work for this version. This represents running the app across multiple systems to allow them to be on teams and play against each other. This would require the distribution to be lightweight API callable through standard network protocols that executes quickly. This would be best developed using object-oriented design principles. RESTful API supports the lightweight functionality for faster processing. This would force the client to provide information needed for the server to fulfil the request and is also callable through HTTP.

Programming Language – The app will be built using Java programming language. Since the existing code is on the Android platform and Java is the official language for Android app development, the code will be leveraged for the web-based solution.

Team Knowledge of Web-Based App Development – The development team needs to be knowledgeable about web-based app development principles to create the optimal user experience, maintainability, and usability.

## [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 four classes at the bottom of the diagram represent associative relationships from left to right. Each class on the left is comprised of objects from the class to the right. The GameService object manages the games presently played. Only one instance of a game can be active at a given time. The Entity class is a base class where it forms a foundation for the other sub classes. The sub classes have their own attributes and inherit the shared attributes from the base class. The ProgramDriver and SingletonTester classes have a directed association relationship shown by the arrow pointing from the driver to the tester. The driver class is the main point of execution. The driver uses the tester to validate the service and ensure fulfillment of the unique game instance requirement.

**"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 Mac operating system can be used as a server. There is a weakness that the licensing is costly. They also have specifics that you have to have Mac books to develop it. | This is the most popular operating system because of it’s cost being free for licensing. This operating system is available for web-based hosting. | The Windows operating system has high security option, flexible, and easy to use/setup. The licensing can be expensive, but it has a server option. | These can be used as servers, but they are not very equipped to be beneficial as a server. They will run slower and do not have the power capabilities required to run the server. There are options for it to be used with development. |
| **Client Side** | This operating system is strict that you must use a Mac book to develop This can be costly. They can use SDK’s, which they are good for though. | This operating system would use python and would need someone experienced in working with python. There is also the cost being high because of development time. | Windows operating system requires a high expertise. They would recommend using .NET framework to have greater security and capability. | With mobile devices, experience with development in apps would be necessary. There are things that need to be done off the web such as displays and user interactions. |
| **Development Tools** | For the Mac operating system iCode would be needed on the Mac Book so that coding can be done using swift. | Linux operating system comes with Python installed and you can use IntelliJ’s Ultimate IDE to be able to code this. | The best way to code with the Windows operating system is Visual Studio. This is also standard. Most Windows programs use C++ or C# for languages, but you could use any language theoretically. | Android operating system requires someone that specializes in Android Studio for the app development. With iPhones, a Mac book is needed to use swift in iCode. The other options is that it could be coded into C++ to be converted later into an android app or iPhone app. This is called unity and a Mac is still needed to convert it for the iPhone. |

## 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**: Windows seems to be the best operating platform to use with this game since it is widely used and most people know how it operates. There are many options for the IDEs to be utilized with Windows, and this operating platform’s costs can typically be lower overall.
2. **Operating Systems Architectures**: For the Windows architecture, it allows software applications to use the power of Windows to have a GUI/window set up and allows access to memory and other processes without impeding other operating platform work.
3. **Storage Management**: Windows has storage management built into it through Disk Management, Disk Cleanup tool, and Storage Sense options. Disk Management is utilized for advanced storage tasks and the others help maintain storage on the system by disposing of the unnecessary files that are only taking up space rather than being functional.
4. **Memory Management**: Windows also has memory management built into the system which means that there would only be a database that is needed for all the game’s image files. Having the database will allow the application to use the files to be easily accessible when they are needed.
5. **Distributed Systems and Networks**: It would be best to utilize a client-server distributing system as each client depends on their own server application for the game. The game would need to have the capability to have multiple clients access one server to play the game together. Given this, the server network needs to be strong.
6. **Security**: As previously mentioned, Windows has an option for security with the Windows Defender included within its architecture and features. This would need to be encrypted for the data to be sent back and forth.