

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)4

[**Evaluation**](#_2o15spng8stw)4

[**Recommendations**](#_m8aleynsvzvc)6

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 07/15/22 | Cristian Chavez | Additions to: Executive Summary, Design Constraints, and Domain Model |
| 1.1 | 7/30/22 | Cristian Chavez | Additions to: Evaluation |
| 1.2 | 8/13/22 | Cristian Chavez | Additions to: Recommendation |
|  |  |  |  |

**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 wants a web-based version of their gaming app named “Draw It or Lose It”. This game application will be developed in Java and will be able to host a game with multiple teams involved, with the ability to have multiple players per team. Unique identifiers will be implemented to ensure there are no issues with memory use.

## [Design Constraints](#_2et92p0)

* **Ensuring game application can be accessed on different browsers**

Unless The Gaming Room wants to require/recommend the use of a specific browser, all the resources/plug-ins should be available on the web page so that the browser that is being used to access the game is not important.

* **Ensuring web-based version is similar to original gaming app**

To ensure that the users of the gaming app are receiving the same experience on the web version, it should be a similar as can be to the original application

## [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 class “Entity” is the Parent class that the classes “Game”, “Team”, and “Player” will inherit attributes from. Inheritance and polymorphism are used to allow classes to have the same variables without adding extra code, along with allowing a variable of the same name to be changed depending on the class.
* “GameService” will be the main class and it associates with the rest of the classes, it is able to have 0 to many Games, which a Game is able to have 0 to many Teams, and a Team will have 0 to many players.

**"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** | Most Macs, the ability to add server capabilities can be done through Apple with macOS Server for $19, on the Apple store. Very simple to set up and can be upgraded to handle more traffic as needed. | Is open source, so no licensing costs are required. Is extremely flexible and allows for certain scripting languages to be used for cloud server hosting that other OS don’t allow. High performance. | Windows is an extremely common OS, a license to fully use the software is required. Easy to launch with all required components at the start. |  |
| **Client Side** | Macs are expensive, however they are relatively easy to navigate and work with so a large time commitment is not necessary. | Linux is free to install, however has a steep learning curve and not as user friendly. There is a large community available to help If dedicated enough. | Devices that run windows aren’t too expensive and are easy to navigate. Beginner friendly and have helpful UI. | For mobile devices, there are various amounts of changes between each phone and would be from various manufacturers, so they would likely benefit Client-server pattern so that there doesn’t need to be multiple versions for each phone. |
| **Development Tools** | **Languages:** Swift, Objective – C, Flutter, C#  I**DE –** Xcode  **Tools -** GitHub | **Languages:** C++,C, Java, PHP, Perl, mySQL, JavaScript, Shell  **IDE –** Eclipse, NetBeans  **Tools -** GitHub | **Languages:** C#, JavaScript, Java, SQL, PHP  **IDE:** Visual Studio, Eclipse  **Tools -** GitHub | **Languages:** JavaScript, PHP, Ruby, Java  **IDE:** VS Code, Android Studio  **Tools:** GitHub |

## 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 appropriate operating platform for this application would be Windows. While there are costs associated with licensing the Operating System, the ease of use of the system and familiarity with it will allow for easy maintenance.
2. **Operating Systems Architectures**: There is a limit of 20 connections put in place for a server ran of a Windows device, so Windows Server would need to be used since it doesn’t have a connection limit. Using Windows Server gives access to Azure, which is a cloud based computing service, so The Gaming Room doesn’t need to provide all the processing power themselves and can opt for a hybrid approach between On-premises and Cloud.
3. **Storage Management**: The data on the servers is important to the client’s business and should be safe in the case of a failure on either side of the server. Storage is kept in tact through storage redundancy, where there is a local copy that’s of course kept on premises, and then a “zone” copy that is kept on the cloud. By putting that data in more than just one safe place, it is made easily recoverable if something happens.
4. **Memory Management**: Windows uses Virtual Address Spaces for referring to processes in memory. There is also the use of cashed storage so that the memory can refer to that saved data faster than it would if it was fetched from the hard drive.
5. **Distributed Systems and Networks**: While using Windows Server, another platform such as Mac can connect to the server as long as they have a version of their own platform that is reasonable recent, and they have an internet connection. Windows server provides a Connector software that would communicate to the Mac device and allow for the connection.
6. **Security**: Window Server uses encryption for all connections to server using AES-256 encryption. In the event there is a malware attack, the Operating System itself does not have access to user credentials and keys since they are stored in a secure place using Virtualization.