

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 |
| --- | --- | --- | --- |
| 1.0 | 12/17/2023 | Hayden Gentry | Requirements, Constraints, and Evaluation |

**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)

Our client, The Gaming Room, has consulted with our company to develop a web-based game that serves multiple platforms based on their current game. Their game, Draw It or Lose It, is currently only available on Android platforms. The proposed solution is to create a Java program built upon OOP principles. The program will include separate classes to deal with managing the information of the game, teams, and players. Using a single class to interact with the other classes to ensure encapsulation. This design approach will successfully meet our client’s business and technical requirements.

## Requirements

*<* Please note: While this section is not being assessed, it will support your outline of the design constraints below. *In your summary, identify each of the client’s business and technical requirements in a clear and concise manner.>*

## [Design Constraints](#_2et92p0)

* Each game must have one or more teams.
* Each game must have multiple people.
* Game and team names must be unique.
* Allow users to view take team names when choosing their team’s name.
* Only one instance of the game can exist in memory at any time.
* The game must be able to run on multiple web-based platforms.
  + Must work over the internet. Dealing with multiple live users, network traffic and security issues. Considering the time constraint is a major component of the game, latency is a huge consideration for the users.
  + Follow OOP design principles. Making the program easier to use and maintain. This is important for our clients as games often go through many updates and expansions throughout time.
  + Handle a growing base of users. Dealing with server-side issues like dynamic network traffic and resource management. As the game becomes more popular, we cannot see a dip in game performance handling more users.
  + User friendly UI across platforms. More effort into making the program visually pleasing. The game should be engaging for players and the controls should be intuitive.

## [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 Game, Team, and Player classes have an inheritance relationship with the Entity class. This means the Game, Team, and Player classes inherit variables and methods from the Entity class. Making the Entity class the parent class, and the Game, Team, and Player classes are child classes. This avoids redundant code and consistency across classes. The GameService class relationship between the Game class is “has-a” relationship. A “has-a” relationship is where one class has a reference to an instance of another class. Similarly, the Game class has a reference of the Team class, and the Team class has a reference to the Player class. The SingletonTester class uses the ProgramDriver class to test for singleton behavior.

**"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** | Mac hosting is popular for hosting specifically macOS.  Advantages: Easily upgradable for different requirements  Disadvantages: Less preferred for hosting | Linux is known for being open source.  Advantages: Cost-effective and secure. Most preferred for hosting  Disadvantages: Difficult to find supporting applications | Windows hosting is popular throughout the business world.  Advantages: Many supporting applications  Disadvantages: Plagued with security issues | Mobile devices becoming popular introduce many to mobile gaming  Advantages: Rapidly becoming more prevalent. Portable  Disadvantages: Compatibility issues with the variety of mobile devices |
| **Client Side** | Cost: Moderate, similar cost to Windows.  Time: Would take longer to implement than Windows  Expertise: Hard to learn. Not widely used so difficulty to get support. | Cost: Cheapest  Time: Most amount of time needed to implement  Expertise: Hardest to learn, many functionalities and not the best UI. | Cost: Moderate, similar cost to Mac.  Time: One of the faster options  Expertise: Easier to use than most other hosting options. | Cost: Most expensive.  Time: Would take longer to implement than Mac, but not Linux.  Expertise: Harder than Windows. Different platforms and devices to consider. |
| **Development Tools** | Programming Language: Swift  IDE: XCode | Programming Language: C  IDE: Eclipse | Programming Languages: Java, Python, and C++  IDE: Visual Studio | iOS  Programming Language: Swift  IDE: Xcode  Android  Programming Languages: Java, Kotlin  IDE: Android Studio |

## 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 operating platform that I would recommend The Gaming Room is Windows. Windows has many different applications available while a having a good compromise between cost, time, and expertise.
2. **Operating Systems Architectures**: The Windows operating platform provides many different applications that allow the access of to all characteristics of the architecture we are creating. Some of these applications include graphics, web services, and system resources. These applications and others can be accessed through GUI’s which makes it easier for novice developers.
3. **Storage Management**: Depending on the version of Windows operating platform you use, will determine how your storage is managed. With Windows 10 and above, have a built-in feature called Storage Sense to manage storage. The primary feature is handling where files are in storage and how much storage they take. With other features like recovering deleted files, scheduling file deletions, and cloud storage. This built-in feature ensures the workflow of the developer team to work smoothly with many files.
4. **Memory Management**: The game Draw It or Lose It requires to have access to a database of many stock images. Having a library of images that can be accessed outside of the project allows for faster loading of the project while having quick access to the library of images. This is because we only need to access a small number of images at a time and do not need to constantly load the whole library of images. This also enables the team to keep development on the project and image library separated.

1. **Distributed Systems and Networks**: To fulfill the clients request for Draw It or Lose It to communicate between various platforms, we first will expand out their current web server capacity. While this comes at a cost, it is required with the expected increase in traffic from new users. With a Client-Server based model using the HTTPS protocol; the client sends requests to the web server, and the web server sends responses to the client. Similarly, the web server communicates with the application servers, where the web server is the client. Though with all the connections from the client to different servers, has the opportunity for mistakes such as bad connections or outages. Having our servers separated enables troubleshooting only the affected servers while keeping the rest online. This has the benefit of being able to build out our servers only where they are needed.
2. **Security:** The Windows operating system has built in security protections including virus and threat protection. This could be used to constantly check for malware, since new exploits and malware are coming out it is important to always be checking for threats. It should be required to have a login page for the game to verify the user’s identity. The safeguards would include a username and password that could be verified through email, and further secured with two-factor authentication. Windows also has a built-in feature for VPNs which would further help with preventing network traffic from being monitored.