

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 | 01/19/22 | Kerrian | Completion of template |

**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)

“Draw It or Lose It” is a game that The Gaming Room wants to create based on the popular 80’s game show *Win, Lose, or Draw*. The purpose of this game is for different teams to take turns guessing an image that is revealed at a gradual pace. If one team cannot guess the image in time then the other team has a fifteen second opportunity to guess the image and claim a win. Each game lasts a minute, and there are four rounds of games until it is game over. “Draw It or Lose It” will be a web application that Creative Technology Solutions will aid in building. In order for this application to run as intended, The Gaming has provided a list of software requirement.

These requirements are as follows: the game must have the ability to have one or more teams involved. Each team will have multiple players, so it is important to avoid the confusion of teams and players having the same name through programming restrictions. Unique instances, identifications, and names will be a requirement, so players will be alerted if they have made the mistake of trying to run multiple instances or identical names.

## [Design Constraints](#_2et92p0)

In the process of developing “Draw It or Lose It”, there a few design restraints that must be addressed. The first design constraint is the fact that “Draw It or Lose It” is currently an Android only app. The Gaming Room would like their application to become a web-based application that can operate on multiple platforms. This means that operating systems like Windows, Mac, mobile devices, and Linux will need to be compatible with the application. The source code of the game will need to be examined and updated to become compatible with all platforms. As for the game itself, The Gaming Room’s software requirements must also be put into consideration during the adjustment of the source code. There may exist future operating systems that will require “Draw It or Lose It” to continue making adjustments to source codes in order to open themselves to a wider audience of players who are interested in trying out the game. From a business standpoint, The Gaming Room will need to spread word to new and current users that they are now able to play “Draw It or Lose It” from any device that they own. A marketing team will need to advertise the convenience of these new platform capabilities whilst a programming team remains on standby to address any bugs that may come up after the new versions are rolled out. Feedback from the player base will be vital around this time as well; therefore, it is suggested that The Gaming Room’s marketing team have members who can take back the comments and complaints about the software to the programmers for essential updates to be made.

## [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)

Below is an image of The Gaming Room’s UML Diagram for “Draw It or Lose It”. This diagram serves as a map in terms of pinpointing which classes share relationships and the type of relationships they are meant to share in order for this software to run as it should. The Game, Team, and Player classes all have a relationship with the Entity class known as an “is-a” relationship. This means that these classes are a part of Entity and inherit traits from Entity such as id and name. The arrows pointing from Game, Team, and Player towards Entity symbolize the connection that they share in the program. There is also a series of connecting relationships starting with the GameService class and moving to the right to the Player class called a “has-a. So, Game Service “has a” Game, Game “has a” Team, and Team “has a” Player. To the upper left is the Program Driver where the program will run using the SingletonTester class to create a single instance of a game. As mentioned in the software requirements, The Gaming Room would like each game instance to be unique, so the use of the singleton pattern while programming “Draw It or Lose It” is essential.

**"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**

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Mac offers a user-friendly environment to modify the server and web application as needed; An advantage of Mac besides its popularity and accessibility is the ease of connecting to Apple devices like iPhones and iPads; A weakness of Mac is its high costs, especially when it comes to configuring a Mac with the strength to handle tasks like software development. | Linux is highly customizable and secure which could be ideal for web hosting; An advantage to Linux is cost since most of the software is open-source and Linux itself is free; a weakness is that it is not user friendly or as popular as the other OS which means less application support online. | Windows is the most popular OS and has been trusted for decades; An advantage of using Windows besides its strength in handling software is its dependability and nearly endless support; a weakness of using Windows is that its ability to withstand viruses, worms, trojans, and other malicious programs has been a long-time issue. | Mobile devices offer a great deal of flexibility; an advantage of web hosting on mobile devices is the availability of free apps that could aid with hosting web-based applications; a disadvantage is poor security and lower memory. Some websites still need to make mobile versions of their standard sites to prevent mobile devices from lagging or changing layouts to fit on smaller screens. |
| **Client Side** | Mac will require some expertise since, as popular as it is, it is still not a system that everyone is accustomed to using. Macs are also costly, and require Apple experts if hardware support is needed. Macs will need the ability to simulate their program in Linux and Windows, but testing on a mobile device should not be a major issue since transferring data via the cloud makes testing on a Mac and iPhones seamless. | Although Linux will cost less because it is free and offers many open-source programs, the time and expertise required to run Linux and then make Linux simulate other environments can pose a challenge. There is also not as much support for Linux as Mac and Windows, so any application problems can lead to more development and testing time than normal. Testing for mobile devices can pose a great challenge also. | Due to the flexibility of Windows and how common it is for the average household to have a Windows-based device, expertise from the client will not be a major problem, and neither will time. Cost, on the other hand, can be just as high as the Mac but without the ability to test as seamlessly on mobile devices that Mac offers. Finding tools on Windows to simulate other environments for testing should not pose a huge problem though. | Since using applications on mobile devices is a daily thing for most people, cost, time, and expertise would not be a major problem. Flexibility also allows files to be transferred from different mobile devices and OS with relative ease. The biggest problem for mobile devices remains the question of memory and general strength to develop software. |
| **Development Tools** | Mac is capable of running various programming languages, and like most software it has its own Mac version for programming (Swift). Mac can also run Visual Studio and Eclipse. Web development is possible with the listed software, so building a web application with web-based languages like HTML and CSS would not be an issue. | Once more, Linux is very friendly with open-source programs, and there are plenty of tools available to Linux users to build software. Linux can also run the likes of Visual Studio and Eclipse in the same way all OS can: by downloading a compatible version. Since Linux has plenty of software writing programs, it can easily develop applications. | Windows is more likely to be compatible with any software development tool found than Mac, Linux, and mobile. It is not unusual for software to be developed for Windows first then adapted to other OS later. Visual Studio and Eclipse run well with Windows, and developing web applications through Windows is a common method. | Mobile Devices have the advantage of app stores. Building applications for and with Mobile Devices is becoming more popular, so tools are becoming easier to find. The only downside is that mobile-based web browsers are not always compatible with the HTML, CSS, etc. created on standard web browsers, and may move things around unintentionally. It is often better to develop applications via apps rather than web when it comes to Mobile Devices due to this. |

## 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 is Windows. It is not only a platform that offers plenty of development tools and resources to software developers, but Windows has been the most popular method of computer gaming because computers that run on Windows tend to be more powerful than computers that run on macOS, for example. It is user friendly to both the developers and the players, so I believe it would be the best platform to start on.
2. **Operating Systems Architectures**: Windows-based processors runs in user mode and kernel mode when relying on RAM. In user mode, applications like Draw It or Lose It run while kernel mode runs the operating system itself. The 64-bit version is able to handle more RAM usage than a 32-bit, so a 64-bit is recommended. This application will be rendering a lot of photos, so a processor that will not lag while juggling user mode and kernel mode would be ideal.
3. **Storage Management**: When looking at the best storage to copy the files that Draw It or Lose It needs to operate, the recommended means of storage is an SSD (Solid State Drive). Compared to an HDD drive, an SSD drive is faster at accessing data in storage and delivering better performance all around. Every gamer knows that speed is essential to enjoying a game, even when it comes to storage. In terms of the storage space itself, it is always recommended to choose a hard drive that has enough space to copy all of the files that Draw It or Lose It needs run. Certain files that contain content like saved progress or settings can always be backed up in cloud storage in the event that the computer itself becomes inoperable and accessing that progress from another system is necessary.
4. **Memory Management**: It is always ideal to have the latest, largest RAM since it translates to an operating system that can handle more tasks. In order to run an application like Draw It or Lose It, the recommendation for memory management is RAM with at least 16 GB. Having RAM of this size will allow users to play Draw It or Lose it with minimal lag which is vital when playing a game that relies on loading images at a constant pace.
5. **Distributed Systems and Networks**: Draw It or Lose it should run on a client-server distributed system. The first step that The Gaming Room would need to take is having servers ready that can handle the number of users they anticipate having on peak days at peak times. A server overloaded with too many users connected to it will start connection issues for everyone, and even lead to a server-wide crash. Once servers are up to standards and Draw It or Lose it is released, the next step is to establish the client-server network that users will need to play. Once users install the game and it runs, the user should be connected to The Gaming Room’s servers along with every other user who plays the game. Doing this will allow a seamless gaming experience where users can play with one another from all over the world and experience the same things in real-time. On the user’s end, it is definitely recommended that they choose an internet plan that can handle the amount of bandwidth required to play an online multiplayer game like Draw It or Lose It. On the developer’s end, there should be constant management of the server in terms of updates, scheduling regular server maintenance, and making changes to accommodate growing client bases as often as possible. Some MMO (Massive Multiply Online) games establish servers for specific regions of the country and the world in order to make client-server network management more tolerable.
6. **Security**: It is highly recommended that The Gaming Room requires users to log in to the server with at least a two-factor authentication. By having the user verify their identity twice, the odds of hackers breaking into their account is drastically decreased. Once the user is logged in to the server, it is recommended that users have another log in for their Draw It or Lose It account as well. When a user has too many login attempts, they should also be locked out and contacted via email about the excessive log in attempts in the event that they were not the ones behind it. These security measures are not uncommon with MMO games, for example. Users are often required to log into the console of their choice before logging in at the main page of the game. This main page often displays information about security threats and updates that keep users informed as they wait for their account to be logged in. The server will also lock them out after a set period of time if they continue make excessive log in attempts. As for Windows itself, most systems come preinstalled with Windows Defender and firewalls, but there are also free antivirus and anti-malware software available online to provide extra layers of protection. The server itself should maintain a log where unusual activities can be investigated.