

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 | 07/15/2024 | Ryan Blough | Completed Executive Summary, Design Constraints, and Domain Model sections. |

**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 wishes to develop a web-based game that serves multiple platforms based on their current game, Draw It or Lose It. The teams will compete to guess what is being drawn. The application will render images from a library of stock drawings instead of the player drawing the item themselves. The game should consist of four rounds lasting one minute each. The drawings are rendered at steady pace for the first 30 seconds, if the team guessing does not guess correctly after their allotted time, then the opposing team will have a 15 second time limit to provide one guess. There software requirements are the following:

## Requirements

* A game that will have the ability to have one or more teams involved.
* Each team will have multiple players assigned to it.
* Game and team names must be unique to allow users to check whether a name is in use when choosing their team’s name.
* Only one instance of the game can exist in memory at a given time.

## [Design Constraints](#_2et92p0)

* Game must be developed in a web-based language such as Java or Python in order to provide cross functionality between Android and iPhone users.
* Written to allow multiple teams to play at one time, with multiple players on each team by implementing team and player objects.
* Ensuring to check that game and team names are unique using the singleton pattern for the creation of each game and team.
* Allowing only one game to exist at one team in the game’s memory by creating a unique ID for every game, team and player with the use of iterators.

## [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 Entity class incorporates the Game, Team and Player classes, which allows the attributes and operations contained in the Entity class to be used for each of the of the classes it incorporates that it inherits. The four subclasses (GameService, Game, Team and Player) are all associated with each other. They use a zero to many association, which means that can use the instance as many times as it needs to or zero times. The ProgramDriver class uses the SingletonTester class to ensure that the game is functioning properly.

**"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 can be used as a server web-based software application. However, it is not chosen as much as Linux or Windows. It uses a Unix-based environment. Unix-based systems provides powerful security and is very stable. Mac OS server also uses features such as the Apache web server. Some weaknesses for Mac is that there is a limited amount of hardware choices for server-grade Mac. It is also a more expensive choice compared to Linux or Windows. | Linux is a very popular choice to use for web-based software applications. It uses a large amount of distributions that are suitable for server use, it provides great stability and scalability. Linux is very cost-effective, it is very easy to customize, and offers a large range of server software options. Linux is very well known for its security and reliability. A weakness of Linux is that does require more technical expertise to set up and manage when it is compared to Mac or Windows. | Windows is probably the most commonly used platform for web-based software applications. Windows has excellent integration with Microsoft tools and technologies. It offers a wide range of web server options, such as IIS. It also has good support for .net applications. Some weaknesses that Windows has is the licensing costs can get a bit pricey. And it is not really well suited for open-source web development. | Mobile devices are not typically used for web-based software applications. Servers are used to serve mobile app data, and any of the three platforms already described can be used as mobile app backends. It does have a few advantages including its proximity to data for reduced latency. It is also cost-effective for its specific use cases. And it is relatively easy to deploy small-scale apps. There are a lot of weaknesses regarding mobile devices, due to its limited resources, which can cause performance issues, reduced scalability, reliability issues, and security risks. It is reliant on networks that they use and could have potential bandwidth limitations. Also maintenance and management complexities. |
| **Client Side** | Developing for Mac typically involves using Apple’s development tools, like Xcode. The cost of using Mac is a moderate choice, and you may need some expertise in Swift or Objective-C may be necessary. | Developing for Linux can vary depending on the distribution and the environment that the desktop is using. It doesn’t typically cost very much, but it does require more expertise for its distribution-specific considerations. | Developing for Windows often utilizes Visual Studio (as a Microsoft product). Costs can vary depending on what you want to accomplish, but expertise in .net languages like C# is typically necessary. | Developing for mobile devices involves a platform-specific development. Cost vary depending on how many platforms it is implemented on. And expertise with Swift (iOS), Kotlin (Android), or cross-platform tools like Flutter may be required. |
| **Development Tools** | Xcode is the primary IDE that Mac uses for its applications. It also can support Swift, Objective-C, and C++. | Linux has a very wide range of programming languages. Some of the most popular IDE’s that it uses is Visual Studio, IntelliJ IDEA, and Eclipse. | As mentioned above Visual Studio is the most commonly used IDE as it is a Microsoft product. Visual Studio supports a lot of different languages including C++ and Java, but has much more to offer as well. | There are a wide range of development tools for the different mobile devices. For instance, iOS, Xcode and Swift/Objective-C are used. Android however uses Android Studio and Kotlin/Java are more common. Cross-platform tools like Flutter can assist in developing for both platforms. |

## 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**: Linux would be the best choice for developing the “Draw It or Lose It” game. Linux will provide with lower licensing costs and will not limit access to its data centers the way an Operating Platform, such as Microsoft would. It also provides good security and is very user friendly, because it is one of the most commonly used server platforms. It also has many tools that we can use for assistance in development. Using a different programming language for each device is also an essential part of the development process, for example using Swift for iOS or Java for Android.
2. **Operating Systems Architectures**: The Linux Operating System uses what is called the Kernel, which is considered the core section of the operating system. It has the responsibility for every major action for the Linux OS. The kernel runs the underlying hardware and provides the communication between the hardware and the software used for the game. The OS will then proceed to run through a shell utility, which is an interface between the kernel and the user. The shell will take the commands the user input and will run the functions provided by the kernel.
3. **Storage Management**: Storage management for this project will not be very taxing, as it only calls for approximately 2 GB of image files as well as the code to run the program. Either HDD or SSD storage can be used for the project, using SSD storage will provide the user with faster access to the images which will provide the user with a better experience for the game.
4. **Memory Management:** Linux’s memory management utilizes the virtual memory model to map memory addresses from a program to a physical location, such as the server. The kernel is then responsible of allocating and deallocating memory for the physical location. It utilizes what is called the Paging mechanism, which divides the memory into a smaller size called pages, which can then be called upon when needed. Another feature that Linux uses is what is known as swap space, which means that if an application requires more memory than is available in the physical memory, the kernel will use a hard disk as an extension to the server.
5. **Distributed Systems and Networks**: Linux uses what is called Kubernetes which will allow Linux to be distributed on many different platforms. Kubernetes automates different tasks of container management and has built in commands for deploying and updating applications. It also allows the application to scale up or down depending on its needs, as well as monitoring the application.
6. **Security**: Linux is one of the most secure operating systems on the market, they are able to do this a few different ways. User privileges are one of the ways they are able to accomplish this, because all users are required to have individual ID’s and passwords. Additionally, Linux has multiple level of access privileges, and users are automatically assigned a lower access level, which will limit their access to mainly computer files. Linux also uses an open source code which will actually boost its security, and because its source code is split into multiple subsystems that are maintained regularly, it ensures that changes to the code are properly vetted. Linux also tracks everything so that they can easily identify anything from a failed log in attempt to something that is more malicious.