

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 | 01/27/2025 | Kaley Lavery | Revision One, Initial Proposal |
| 2.0 | 02/07/2025 | Kaley Lavery | Revision Two, Evaluations |
| 3.0 | 02/13/2025 | Kaley Lavery | Revision Three, 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)

This software design template is used to show the recommended design approach for The Gaming Room. With this document, there should be a clear and concise understanding of how *Draw It or Lose It* would work, with explanations of the software design provided, anticipated problems and comprehensive solutions to meet the requirements of the application requested by The Gaming Room.

## Requirements

Creative Technology Solutions will create an application that allows the player to engage and interact with the gaming experience across different platforms. These platforms must be accessible by both web-based browsers and multiple devices.

## [Design Constraints](#_2et92p0)

Creative Technology Solution’s goal is to address any potential limitations within the application whilst being able to provide the user with an enjoyable experience. Foremost, addressing browser compatibility to ensure that the application can be used across multiple web browsers, including Microsoft Edge, Mozilla Firefox, Google Chrome, and more. CTS will also address the network latency, ensuring that the application can operate smoothly across an array of platforms and devices whilst considering any browser-specific features and/or capabilities as well as remaining standards compliant. The application will have implemented measures to protect any user data by the means of encryption, authentication mechanisms and methods to validate user information input. Furthermore, providing an application that addresses scalability, or a scalable infrastructure by utilizing caching mechanisms to handle a growing user database and changes to user traffic, while still providing consistent data in a way that is visually appealing and responsive will be on the forefront.

## [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 Domain Model for *Draw It or Lose It* is represented in the UML class diagram to show the game application and its classes and relationships within its singular entities and the overall behavior of the system. First, the entities are the classes that share common attributes and behaviors with the other classes within the UML for the Domain Model, meaning that the other classes inherit the properties and methods form this class. Next, the User is a class that represents what the individual user can do, holding User specific information such as their username, their password, scores, current and passed gaming statistics, and history. This class showcases encapsulation, as you can only manipulate the information belonging to this class by providing the correct information through the security authentication methods. The class Game represents an instance, specifically of a singular game, including the players within the game, the progress happening within the game, and other game specific data. The class Game uses the aggregation principle, as it aggravates player objects. The class Player represents a user that is currently participating in a game, showcasing their scores, statuses, and progress within the game. This represents association by demonstrating the relationship between players or teams and games and show they behave together. Teams are a congregated group of Players that collectively are seeking the same object, but they show similar things such as their team name, the players within the team, and the team score and progress.

**"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 offers Unix-based environment, which is known for being both robust and flexible. While this simple to execute tool is favorable, it is less common to be choose comparatively to both Linux and Windows. However, it does offer Apache web server, which is generally offered through other operating systems as well. Its downfall is the lack of technical or hardware choices, as they are vastly limited to Apple products. | Linux is arguably the most common operating system used for web-based software. It can handle high-volume applications and is vastly adaptable in terms of scalability and overall stability. It is a cost-effective choice but can sometimes be more technical to set up than opposing operating systems. | Windows is another very commonly used operating system for web-based software. It integrates well with its own Microsoft technologies, and offers a range of web server options- Microsoft IIS (Internet Information Services) being the most prominent, as it’s built-in. Apache HTTP server can also be downloaded and used easily- making integration easy. However, the cost of licensing can be rather high and not well-suited for an application that may utilize open-source software. | Web applications aren’t normally hosted on mobile devices, but instead on the servers in which the mobile apps run off of. All of the prior mentions of Mac, Linux or Windows can be used across mobile devices. Mobile devices are ideal for smaller scale applications, but this can limit performance and scalability challenges. Mobile devices are also limited to the battery and hardware that exist in the phone, along with network dependency. |
| **Client Side** | Development for Mac applicants is more complex than developing either Linux or Windows, as Xcode is usually a required developmental tool and this can lead the costs up, as well as the need for someone more versed in Mac based platforms. | The cost for developing Linux applicants is relatively low, as this is a free and open-source operating system. | The cost can vary, as stated with the high cost of licensing, but otherwise can remain relatively low as Visual Studio and expertise in typical C languages (C#, C++) is necessary. | The cost for developing mobile devices is dependent on many factors, including the number of platforms intended to target. The need for expertise in either iOS, Android, and other cross-platform languages could be required and costly. |
| **Development Tools** | The integrated development environment most used amongst Mac is Xcode, although it does support C++ and Objective-C which are both generally acceptable to use cross-platform. | This operating system supports many programmed languages and IDEs where it can be utilized. This is an ideal choice when considering cross-platform versatility. Languages supported include Video Studio Code and Eclipse. | This operating system supports many of the programming languages and IDES where it can be utilized, such as Video Studio Code, C++, and C#. | Utilizing mobile devices as a platform means that there are different scenarios per platform. iOS has Xcode and Objective-C, Android uses Java, and then there are cross-platform tools to utilize such as Flutter or other open source software for cross platform app development. |

## 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**: It is recommended That the Gaming Room chooses a web-based platform when considering the expansion to other computer environments. There is more versatility in how the application can be accessed and a wider range of users when considering a web-based platform. These platforms include desktops, smart phones, tablets, and laptops that are not brand specific.
2. **Operating Systems Architectures**: Web-based operating platforms allows the application to seamlessly operate cross-platform ensuring that users running Windows, Linux or macOS can still use the application. This can be achieved through universally supported technologies such as HTML, CSS, and JavaScript.
3. **Storage Management**: Cloud based storage is recommended as it is reliable, accessible without internet connection, and is scalable. This cloud-based storage can be used to manage the user data and all the related application information using cloud service providers such as Microsoft Azure Blob or Google Cloud.
4. **Memory Management**: Swapping is the recommended memory management because this can move information between the main memory and the secondary memory storage location, when necessary. Automating the allocation and deallocation information and memory storage minimizes the likelihood of leaks or crashes within the system.
5. **Distributed Systems and Networks**: A distributed software architecture needs to be implemented to enable communication between the application and various operating platforms. Utilizing smaller components of the application in ways that operate independently to communicate amongst other small components helps mitigate the risk of fault. The main reasoning being that a distributed system is the implementation of an API, which allows the application to interact with the stored information in the games back end and exchange that data through means of inter-component communication. This eliminates issues such as connectivity and network outages.
6. **Security**: The application should have built-in security capabilities that utilize encryption such as secure socket layers to protect user information on across multi-platform devices. Allowing access controls to ensure that data utilization is limited to those with granted access adds an additional layer to guard sensitive information. Lastly, regular security audits and reactive measures should be taken to ensure the maintenance of a secure platform.