t

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template** 1](#_Toc94723243)

[**Table of Contents 2**](#_Toc94723244)

[**Document Revision History 2**](#_Toc94723245)

[**Executive Summary 3**](#_Toc94723246)

[**Design Constraints 3**](#_Toc94723247)

[**System Architecture View 3**](#_Toc94723248)

[**Domain Model 3**](#_Toc94723249)

[**Evaluation 4**](#_Toc94723250)

[**Recommendations 7**](#_Toc94723251)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | May 21, 2023 | Armando Gomez | First Draft |
| 1.0 | June 4, 2023 | Armando Gomez | Update to the executive summary and revision to the design constraints and domain model explanation |
| 1.0 | June, 12, 2023 | Armando Gomez | Finalize Recommendations based on client’s needs. |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room wishes to embark on developing an online gaming app but has encountered roadblocks in establishing appropriate groundwork for its creation. Several vital criteria must be satisfied before moving forward with this project; these include facilitating numerous teams playing simultaneously featuring several players on each respective side while also ensuring exclusive uniqueness of all game and squad titles so that users can verify their availability when needed. The program's smooth operation demands only one copy of the video game running at any point in time stored within memory. This implementation requires several modifications involving existing framework within pre-established boundaries.

## [Design Constraints](#_2et92p0)

The Draw It or Lose It features an Android-based deployment that is now in use at The Gaming Room. Since CTS has been asked to expand this to the web, the tech stack must support web-deployment; Java has been chosen for this reason. The native Android SDK language, Java, ought to make this new deployment attempt easier.

The Android platform's existing APIs must be examined and/or expanded for mobile use.

## [System Architecture View](#_ilbxbyevv6b6)

Please take notice that while there is no requirement for these projects, this section serves as a reminder that other projects may call for a description of the system and subsystem architecture existing in the program, including physical components or tiers. To comprehend the overall design, a logical topology of the communication and storage components should be presented.

## [Domain Model](#_8h2ehzxfam4o)

When it comes to running the software project within our program framework, the ProgramDriver class plays an important role because it has access to the main method needed to execute most initial instructions. After invoking its public static main method, we have a step that involves obtaining instances of both the GameService and SingletonTester.While calling on GameService itself, it relies on running a private static method called "list games" to collect information about ongoing games. The nextgameid is another useful private static method that holds a unique identifier to each upcoming game. Within the class lies a variable accessible throughout the class - a piece of private reusable software known as "gameservice," which can only have one instance. To avoid other methods from creating additional versions of this service, we include our own private constructor for it.By using a public static singleton method named "getInstance," other methods capable can still access this same variable, and addgame being an example checks for existing games before creating another game if there is no match. Additionally, several getGame methods are in place to help with returning specific games: getGameIndex provides info regarding an index we search for; getGame helps us filter for specific IDs to find their current instance;getGameName serves us in discovering relevant game instances by searching over their names ,while getGameCount just returns how many active games are currently present. Finally, the interaction between GameService and one(or more) available instances of the Game class operates under a zero-to-many relationship .While utilizing Games' own privately implemented 'game' function that hides its default constructor, a publicly available constructor creates specific identities —unique ids and associated names—for current games. One such example is getId() that provides data on each object's id and related content. The a forementioned software system comprises four integral classes; namely Team Class, Player Class Entity Class and SingletonTester Class.

Team Class houses crucial functions such as carrying Team ID and Name records inclusive of: getID (to retrieve Team ID), getName (to retrieve Team Name) & toString (merging both into string format).

Player Class on other hand forms details regarding varying association: from none up till several players within said teams through pre-existing directions detailed via entity inheritance for retrieval mechanisms for getting Player-ID(PlayerID) & Name(PlayerName) whilst utilizing toString function present.

Further mentioning Entity inclusion here– mentioned Game (inclusive of all integral gameplay applications needed), Player/Team-Inherited (remains primary source holding necessary details within variables under one umbrella). This utility ensures there are no wastes resulting due to redundancy/separate control, but all values needed throughout game runtime are easily accessible.

In essence, SingletonTester works hand-in-hand with ProgramDriver class to ensure smooth testing operations occur on the main application program.

**"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)

There are a number platforms that could be developed for, such as hosting/serving or serving as the client, such as Macintosh, Windows, Linux, and mobile platforms. The advantages and disadvantages of each platform are discussed in the table below.

It should be made clear that there is no connection between server and client preferences. That is, if Linux was selected as the server OS, Windows might still be the client's first preference.

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Opting for Mac computers as a go-to solution might seem financially costly at first. However, this cost provides unmatched benefits sure to supplement your computing experience—their impressive stability and performance top even some industry-standard Linux OSs.  The user-friendly interface on these systems allows consumers from any level of computer expertise & responsibility- ranging from home-users to corporate settings- easy navigation and increased productivity. It is worth noting, however; that scaling might present various challenges without in-depth knowledge regarding the device's workings.  The biggest downside lies in software accessibility plus limited Hardware options; nonetheless, with its superior hardware & security features including firewalls—Mac OS X offers excellent data protection abilities against malicious attacks in comparison to competitors.  Finally, easy integration with various applications sets in motion a seamless interactive experience when developing basic web applications since most hosts run Apache servers emulating Linux simultaneously. Besides boot camp setup perfections providing dual operating systems comparable disadvantageously lesser than its alternative equivalents - windows-based platforms exhibit higher complexity requirements such as virtual machines altogether. | The free and open-source alternative would undoubtedly prove to be beneficial in terms of cost efficiency as well as stability and performance quality. With fast and convenient access to FTP tools, users can rest assured that they have taken appropriate cyber security measures since LINUX is arguably one of the strongest secure systems available. Although Linux applications may appear more cumbersome due to their terminal-based operations, this system undoubtedly offers better security than various other unsecured alternatives available in the market.  However, there are some compatibility concerns within certain user groups due to its limited compatibility with Windows’ applications making it difficult for companies reliant on such apps hence posing inconveniences when using Windows on servers. Linux does support a lot of scripts nonetheless migration issues might appear during transition periods. | When considering the numerous historical issues present within various computer systems such as blue screens, system crashes and bugs; it’s vital to note the higher cost affiliated with Microsoft's Windows operating system. Despite its limitations in functionality when compared to alternative options available today; Windows’ easy-to-use GUI (Graphical User Interface) coupled with its quick FTP tools make it a popular option for many users worldwide.  One significant advantage for Windows users is that the platform offers an array of hardware components and software providers - making personalization very accessible; however - these advantages come with a variety of updates/upgrades schedules affecting uptime - resulting in slower load times.  Windows' affordability is an added benefit when comparing the costs associated with usage versus operating systems like macOS but tends to be more expensive than Linux-powered alternatives. Furthermore, right compatibility will dictate your streaming experience when playing Xbox games amongst others hence why windows-based devices tend to work best whereas MACs or Linux servers may fall short due to communication problems between devices causing lagging issues amongst other connectivity-related issues. | Various android mobile applications allow users to code and run servers straight from their phone or tablet. This function is advantageous in its low cost and user-friendly approach however it generally operates best with cloud hosting services. Users face potential obstacles such as connection errors, server configuration complexity, limited hardware capabilities (such as CPU processing power and RAM). And security factors when operating in this manner. Therefore, it’s recommended for smaller jobs rather than large scale ventures like expansive businesses or organizations. Jailbreaking ones phone may grant access to additional functions beyond what’s normally permitted by the manufacturer which adds some flexibility to the process; however one noticeable downside is a drained battery life due to increased demands on resources. With regards to software development separate designs will need to be made for each device you wish to target; nevertheless, choosing an android platform does guarantee larger outreach potential whilst simultaneously minimizing costs associated with upgrades. |
| **Client Side** | A uniform and simplified user interface can significantly improve the overall usability of all Mac devices. Having a singular ecosystem promotes seamless integration between various Apple products such as the Mac and iOS devices made by Apple Inc.. Additionally. Adopting an intuitive graphical interface optimizes the overall user experience by making it easier to use for users of various skill levels with different needs. Time after time research confirms that users find such interfaces significantly superior than those that aim for complexity without clarity (cite). Ultimately this approach holds potential in offering greater protection against cybersecurity threats relative to other alternatives available in the market today. | Python, Java, or C/C++ may all be the most popular and favored languages for Linux development.   Both openness and affordability are defining characteristics of this software since it relies on being free/open source rather than proprietary licensing structures common among competitors. Whilst mastery can seem arduous without requisite expertise at first glance; ease-of-installation protocols maximize accessibility after acclimation period concludes regardless of operating system/browser selection'. Plus, many reliable feature-rich alternatives exist which align perfectly with user demands irrespective of cost concerns. | While acknowledging that other available solutions on the market do offer more integrated options when it comes to cross-platform workflows integration without any noticeable deformity in performance, potential users could still perceive this platform's elaborate functionality flow relatively difficult initially .Despite such challenges though ,it is still worth noting that accessibility is quite satisfactory thanks to simplified navigation inside system's user-friendly dashboard. Crafting bespoke functionality tailored workflows according to desired preference specifications involving hardware capacities can easily be met due to platform's variance in available customizations. Noteworthy inclusions of impeccable gaming options go beyond mere entertainment value and have been specifically handpicked to offer premium graphics for the elevated sensory experience. | The benefits of having a portable device with such convenience cannot be ignored; however, one must consider its limitations like having a smaller screen size which makes implementing multiple features on a single window challenging at times. Moreover, there are limited functions available offline. Nonetheless, there are several perks of using such devices with personalization being among them. In addition, given their portable nature and probability of disruptions; applications should incorporate equipment designed for optimal interruption prevention. |
| **Development Tools** | Mac use Objective-C and SWIFT for development languages.  XCode is the common IDE used for Mac development.  XCode is listed as $99 USD per year per developer. | Linux development may take the form of C/C++, Java, or Python.  Python IDEs, like NotePad++, are frequently free. An additional well-known Python IDE is PyCharm.  C/C++ IDEs are numerous – but not all are available for Linux. Eclipse can do all of these and is free. | Windows is primarily developed using C# and primarily .NET.  Jenkins, TestComplete, and other integration features are available in Microsoft's Visual Studio, a very well-liked IDE.  Visual Studio ranges from $45 – $250 USD per user, depending on features, per year. | The most popular Android IDE is Android Studio, which was created by Google as the official development tool. The Android SDK is Java-based. Downloading Android Studio is cost-free.  The Objective-C and SWIFT programming languages for iOS are virtually entirely created in XCode.  XCode is priced at $99 USD for each developer per year. |

## 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**: As part of our recommendations for The Gaming Room, consider opting for a Linux-based server platform as your intended choice - you won't be disappointed! Its open-source nature coupled with remarkable administration architecture provides top-notch security features while being exceptionally flexible enough to accommodate various software types seamlessly on many hardware setups.

Lest we forget about the incredible resources committed developers provide through their continued involvement in extending functionalities and providing extensive help requires little goading - truly impressive stuff! With large scale events drawing closer daily alongside plans pushing Draw It or Lose It deeper into untapped markets quickly becoming reality; its peerless ability to scale perfectly demonstrates why Linux remains king in many server environments.

1. **Operating Systems Architectures**: At its core, Linux follows a monolithic kernel architecture that operates within the kernel space granting immediate hardware access capability to all orders of software processes attached to it; thus, making for an OS environment that provides top efficiency and performance standards.

However, it doesn't solely stick to this configuration: it extends itself even further with capacity for additional loadable kernel modules established by users themselves - components which could be added or withdrawn from operation mode anytime without necessitating total rebooting of systems. But this feature has significant benefits: it offers better modularity and customization options when dealing with specific needs - like those while gauging support needed to run Draw it or Lose it software.

1. **Storage Management**: It's critical to plan out your company's storage-related needs effectively, which entails taking into account both local as well as cloud infrastructures in the form of hybrid models.

To begin with the Local solution; utilize certified Linux-based compatible file systems namely ext4 or XFS ensuring efficient accessibility when processing high volume data requirements promptly. In comparison, Cloud solutions provide ample scalability backup options maintaining unshakable dependability making them ideal for archives purposes too . We suggest integrating services from reputable providers such as Amazon S3 or Google Cloud Storage which in addition also support functionalities like auto-scaling perfect when tackling varying amount of disparate data volumes.

1. **Memory Management**: Operating on a strong virtual memory management system, Linux aims at curating large and seamless arrays of storage with complete abstraction from actual physical storage. To achieve this goal, the operating system takes use of optimal page division methods allowing smoother processing of program requests.

By adopting efficient measures such as swapping and paging protocols, Linux manages its available RAM optimally freeing up much-needed resources by transferring lesser-used pages onto secondary spaces on disks.

.

1. **Distributed Systems and Networks**: To support cross-platform communication in the Draw It or Lose it game, it might be useful to design it as a distributed system. This approach involves hosting the game server on Linux while developing individualized clients for each targeted platform. HTTP or WebSocket protocols facilitate communication between these clients and the server through network channels; however, establishing solid practices like load balancing, failover and data replication minimizes network outage risks while staying highly available.

By deploying multiple game servers in different locations coupled with smart workload allocation through load balancers ensures continual operation even during failures.

1. **Security**: Online game security is paramount; these demands adopting robust fencing mechanisms at every level of operation on Linux servers tasked with running it successfully; several requirements are necessary throughout its framework to maintain confidentiality and privacy guidelines on active accounts:

At the operating system leverages SE Linux functionalities or App Armor tools effectively grants system-level oversight limiting apps' resources utilization prerequisite offerings from network firewall protective services or IDP/IPS support providers;

Robust encryption facilities employing dm-crypt/LUKS heavily fortify information flows in transit or storage and strong user authentication requirements embedded on systems with adequate password policies and PAM control frameworks and imposing updates regularly to block well-known vulnerabilities.

Strengthening these primary operational frameworks with strict controlled access measures ensures sensitive information remains protected; multi-factor authentications reinforce user login processes, secure practices education campaigns informing users of potential hazards associated with outdated credentials, clicking unknown links, or sharing passwords.

Lastly, having an incident response plan that outlines every step for early identification of breach sources promptly responding to mitigate risks isolating breach impact extent and system recovery plans while alerting affected consumers helps preserve goodwill while preventing a similar issue proliferation in the future. Critical factors at The Gaming Room are comprehensive in their adaptation procedures toward diverse platforms having "Draw It or Lose It" software offerings ensuring all online player data gets optimal protection during gameplay.