

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 | 09/22/2024 | Rolesshania Jackson | Client is requesting OS Recommendations that will be compatible to their specific requirements from budget to fulfill ability for multiple players |

**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 following client has created a game that is like the 1980 Game Show “Win, Lose, or Draw”. The objective requirements client’s game is using an application that will render images from a wide range of library stock drawings as clues, limiting to 4 rounds of play time lasting 60 secs each, and while render is maintained at steady pace with 30 sec marks, the team will have their chance of guessing correct before time expires. When time expires, other teams will have an additional 15 secs to guess/ solve the puzzle. The focus will be to produce a “How to set up the environment” for the staff and simplify the development of the web- based version of the gaming app and streamline the growth.*

## Requirements

*Clients will need a specific outline of what is needed for training for the staff to be brief on for the application development of the game such as:  
\* game ability to have one or more teams’ involvement.  
\* Multiple players for each team (assignment)   
\* Real- time filter for team/game names to ensure no duplication when choosing names  
\* One instance of the game can exist in memory at any time. (accomplishing unique identifiers for each game/team/player)*

## [Design Constraints](#_2et92p0)

<Identify the design constraints for developing the game application in a web-based distributed environment and explain the implications of the design constraints on application development.>

## [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)

This diagram consist of six classes:   
\* ProgramDriver main()- entry point for the application  
\*SingletonTester testSingleton ()- used to test implementation of the GameService Class  
\*GameService (attributes) List<Game>, List<Team>, List<Player>, (methods) addGame, addTeam, addPlayer, getGame, getTeam, getPlayer- management of collecting of the games/team/players  
\*Game gameId (long), gameName(String), adminName (string)- associate with TeamClass  
\*Team (attributes) teamId (long), teamName (String)- aossication with both Player and GameClass  
\*Player(s) (attributes) platerId, playerName  
The relationship associated with GameService is Game, Team, and Player, indicating it managing instance of those classes. Game and Team relation suggesting game can have multiple teams. And Team and Player indicate the team can have multiple players. The diagram demonstrates the following OOP principles:   
\*Encapsulations (each class having its own data and behavior, ensuring internal state of objects is protected.  
\* Association: demonstrating the relationship between classes and their interactions  
\*Single Responsibility Principle: Single responsibility for each class. In regards of efficiency of requirements, we will need to list clear responsibilities (each class responsibilities, easier to maintain), modularity (allowing for individual components to be developed) and Scalability (ensuring certain components are manageable and scalable.

**"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 must 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** | Macs seem to be compatible when introduced to certain web-based software’s. Like Linux, severe to server-side development. | Servers are reliable and versatile. Services include file sharing, web hosting, database admin. | Windows offer great compatibility with game applications but faces high licenses, unless downloading with purchase or compatible with certain models. | Mobile Devices are acceptable to many web-based software’s and at times, offer to download applications |
| **Client Side** | Mac may charge for downloading, use some GB for the space and at times slow down certain model products. Offering stability and smooth performance, especially with Apple hardware. | Linux doesn’t require as much hardware as Windows. Linux is free and reduces licensing costs. Outstanding performance and minimal system crashes. | Windows may charge for downloading, using some space but are prompting users (if any) free up space, but overall will not affect the functionality of model. High operational and licensing cost. | Mobile Devices are quick and easy access to downloads and update on software’s, taking a few minutes than starting back up. |
| **Development Tools** | Mac users mostly use for business and Mac products for outlining business, better visual for users who are looking for better support for outlines | Problems with support and troubleshooting. Fragmentation across Distributions. Less exposed to security threats. Preferred choice for server management and web development. | Windows may have better business backgrounds and compatibility, but for a price. Windows supports a wide range of software and development tools, ideal for programming. Game developers are in favor due to compatibility with other game engines. | Android Development tools are essential resources for creating applications for mobile devices. Devices are open to a lot more options such as IDE (eclipse). Ability to write in more than one programming language. Allowing developers to manage individual software component files. |

## 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**: After further observations of all operating systems, Windows has demonstrated excellent support of software and hardware adaptabilities to perform. This will execute what the clients are looking for and satisfy their goals. Covering how low the cost will be, the expansion of the Game to other operating systems, outsourcing and ensuring compatibility to servers is executed, again, cost demand low as possible to reach a high number of possible demand or customers,
2. **Operating Systems Architectures**: The Windows most importance architectures are the amount of memory can address. Assuming the OP will be the most improved and available version & software, the advantages of Windows have wide support of programming and gaming support. Looking at the server- side architecture, windows server supports in roles like file services, web hosting, etc. This type of design manages and handles high end level applications, databases, and network resources. Windows provides a scalable kernel while managing heavy workloads and high available features, when this is offered to other customers/clients. The client-side architecture offers features like Windows Explorer and various multimedia components. The focus is on security, performance, and compatibility for other applications running on various desktops or laptops. This form of architecture is appealing to other customers who more simpler ways of gaming, Windows client side offer modern interaction methods (i.e. touch, pen input). All differences can cater to the needs of servers such as high availability, scalability, and performance. With clients, you can use a great deal usability, interactivity, and personal computing.
3. **Storage Management** Windows offers several tools and features to ensure storage management. With a built-in utility that allows the user/customer to create, format, resize the client’s storage devices, in other words disk management. Along with storage management, the system automatically frees up space by deleting temporary files, for example Recycling Bin, and other items that are not being used in certain amount of time. Windows prompts users to use OneDrive (cloud storage) to save files online and access them from other devices. A lot of OPs emphasize performance and avoiding keeping what is not needed becomes standard work, defragmenting and Optimize your Drives rearranging and prioritize to reduce slow performances. In situations where the system goes down (temporarily), Windows suggests saving and backing up files, & restoring once system is back up and running. Clients can benefit from an affective backup and restore plan for customers while ensuring that any slow performance from the Game Application will be limited. Client will be exposed to great key techniques and recommendations, management of storage sense, automation of clearance of free space, can manually cleanup disk, and options for backing up system or specific files and restoring them in case of data loss. For client suggestions, regularly back up their system and what other important files they are in possession of to ensure data integrity and availability. Strict schedule for disk cleans up, ensuring the removal of unneeded or unnecessary files and keeping system organized. Although keeping a clean disk will keep an organized system, running tools periodically can assist in performance of clients/customers and keeping keep all systems efficient and safe. Happy client, Happy Customers.
4. **Memory Management**: Windows memory offers several great additives and settings designed to optimize memory for a smoother gaming experience. Windows game mode offers great features that will prioritize by allocating more CPU and GPU resources the game, while overall recovering the game performance. Another feature is memory integrity, which assists in prevention of malicious code from running. Great option to turning it off for better experience and performance. Current clients want to ensure that the performance of this game is nothing less than impressive. Pinpointing the goal to reach other OPs with the game, a lot more consumers would like to make a decision that will not affect their server but benefit by having a great performance while playing. Memory management plays an important role due to customers saving their progression.
5. **Distributed Systems and Networks**: Windows offers seamless communication and resource sharing across multiple OPs. Using DFS (Distributed File System), allows clients to distribute and replicate important files and locations. DFS provides high availability, load balancing, and reduction of network foot traffic. Making file replication between different sites, promising for users to always have access to the latest data. Clients should always monitor the number of virtual foot traffic for the Gaming Application to delegate across multiple servers to continue to improve performance and availability. Windows allows network-based installations, this will be helpful for deploying this system across multiple machines branching out in a distributed environment. Client-Servers communication covers request – response, effectively having clients sends request to the server, wait until full process and sends back a response. Recommendations for clients are understanding Windows communication, using the built-in security features for authentication, encryption, and transport security ensuring solid communication channels. Understanding high volumes of messages, covering scalability. Adaptability to non-Windows systems through SOAP protocols. Building a solid frame for future distribution of systems and networks can fit specific needs through various configuration.
6. **Security**: Windows offers Antivirus that provides real-time protection against malware (i.e. Ursnif), spyware (i.e. Pegasus), and viruses (i.e. Morris Worm). Antiviruses uses a cloud-based delivery system to update with the latest threats. The benefits of Windows firewall and Network Protection, this controls all in and out foot network traffic based on applied rule set, effectively blocking unauthorized access. Clients must look at account protection for customers, credential management such as sign in, fingerprint, number only sign, etc. In other words, the usage of biometric authentication for customers/clients. Since this gaming application will be distributed across other platforms, the importance of protection for app/browser controls warns users against malicious websites, downloads, and apps. Offering effective protection settings to safeguard against vulnerabilities. With data in transit, clients will need to look at encryption while using Windows and network security. The usage of Windows Defender firewall will help monitor and control incoming and outgoing network traffic, while blocking negative activities. Data at Rest, the most reliable is BitLocker, which is an encryption tool that encrypts the entire drive, making unauthorized usage confused about unreadable information. With layers of security, data remains confidential, integrity is kept, and available during transit and stored.