

Draw it or Lose it

# **CS 230 Project Software Design Template**

Version 1.2

## 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 | 09/19/20 | James Heikkinen | Initial document |
| 1.1 | 10/04/20 | James Heikkinen | Requirements |
| 1.2 | 10/18/20 | James Heikkinen | 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)

The client wants to have a game application where it allows multiple teams and multiple players assigned to each team. Every name for each game and team has to be different, which means no duplicate names. Using the iterator pattern and the singleton design pattern will provide a solution to search for already used names and prevent a duplicate from occurring (Lavieri, 2019). In addition, each game, team, and player must have a separate unique identifier.

## [Design Constraints](#_2et92p0)

There are several design constraints to consider when developing this game application. First one, is utilizing the right memory management technique to allow only one instance of a game to exist in memory. The second constraint requires picking the right operating system and programming language that will work well with a web-based infrastructure. Another constraint to consider is ensuring the client knows how the features work by implementing a training session for them to utilize the software effectively. The last constraint to be concerned with is choosing the design patterns to effectively assign a different unique identifier to each game service, team, and player. In addition to this, the design patterns must be able to detective if game service and team names are already being used. Solving these design constraints are imperative for success of the application.

## [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)

In the UML diagram, there are several classes that are involved. The game, team, and player class inherit variables and methods from the entity class. Game class has a bidirectional association with the game service and team class. While the player class only has one bidirectional association with the team class. Symbol “0…\*” means that there can be multiple instances of a game, a team, and a player. Which means, that the UML diagram conveys having more than one game sessions, teams per game session, and players per team. The program driver is the director of how to execute the entire program and has a unidirectional association of the singleton tester class. Overall, this UML demonstrates how the Draw it or Lose it application allows multiple games, teams, and players, as long as there are no duplicates of game service and team names. In addition, each game, team, and player will have their own unique identifier assigned to them (Lavieri, 2019).

****

## [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** | Current version of a MAC OS server is the macOS server. It is a server app that costs about twenty dollars, which is fairly cheap. For an app that you simply install on your PC, it has a ton of feature for profile management, group policy restrictions, and network management. Major downside is this OS is not as sophisticated as the Windows and Linux platforms when it comes to configuring the server (Apple Inc., n.d.). | This operating system has very little demand on hardware, is free of charge, and is least likely to be attacked by cyber criminals. Major downside is the complex operation and updating process of the Operating system. In addition, not all professional software is compatible with the Linux server OSs (IONOS.com, 2020). | The Windows server operating system is user friendly and more intuitive. Easy to install updates, new drivers, and almost all third party applications. Major downside is the cost for licensing, is most targeted type of OS for malware, and uses a large amount of resources (IONOS.com, 2020). | Android does not have an OS specifically for server use. The mobile applications on any mobile device would have to rely on a server from the MAC, Linux, and Windows operating systems. |
| **Client Side** | MAC operating systems are the easiest to use compared to other platforms and take less time to configure and setup. Cost of the OS does not really matter because it comes with the purchase of the PC (Gewirtz, 2019). | Linux can be the cheapest due to being open source. There are many different OS versions of Linux to choose from, so due diligence is a must. Overall operation of the OS is very complex and takes longer to learn than the other platforms. In addition, the OS is a very low target for attackers to get malware onto your PC (Gewirtz, 2019). | With Windows machines and operating system, this platform has the most options for customizing the hardware and software configurations. This is the most recommended platform for gaming and has the most compatibility with most applications. Huge downside to this OS is having the most security risk due to being a highly target OS for cyber attackers (Gewirtz, 2019). | Primary use of these operating systems relies heavily on touchscreen capability. OSs require applications to fit the mobile device’s screen and to be functional with the smaller screen. However, they are much easier to use than a typical operating system on a desktop. A plethora of applications on both Android and the IOS store, but these are the only apps compatible with these platforms. The cost of the OSs comes with the price of the phone or tablet (Ismail, 2020). |
| **Development Tools** | Major programming languages used for the MAC IOS are Objective C, Swift, and Java (SNHU, 2020). APPCODE IDE created by Jetbrains has the capability of using the programming languages of Objective C, Swift, and many other languages (JetBrains s.r.o., n.d.). | Linux mainly uses Java for a programming language for its software (SNHU, 2020. Highly recommended to utilize the Eclipse IDE for programming in the Java language (Eclipse Foundation, I., n.d.). | In the Windows platform, it uses the programming languages C# and/or Java with the .net framework (SNHU, 2020). In addition, Windows has NTFS and Active directory services to apply security to the software. A good IDE to use for this OS is visual studios (Microsoft., 2020). | Depending on the operating system of the mobile device, the safest bet is Java for programming an application. For Android operating systems, using the Android Studio for coding in Java is recommended (Google, 2020). iPhones need the APPCODE IDE for developing apps (JetBrains s.r.o., n.d.). |

## 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 best operating platform to allowing expanding the game environment to all other platforms, is the Windows server operating system. This platform is easy to setup and maintain with security patches. Windows is compatible with most third party applications, which is beneficial for having flexibility for hosting multiple platform environment (IONOS.com, 2020). In addition, it can use the java programming language via eclipse and various other IDEs that are supported by Windows (Eclipse Foundation, I., n.d.). Overall, this platform would reduce any likelihood of compatibility issues.

1. **Operating Systems Architectures**:

The Windows platform adds various other features that are beneficial for a server platform. One major feature is the permissions management. Enabling permissions can limit the access each user has to the system and increase security to protect sensitive information (Silberschatz, Galvin, & Gagne, 2009). Another feature that is useful, is the user interface of the desktop and start menu. It is intuitively designed and easy to understand, making navigating the operating system simple (IONOS.com, 2020).

1. **Storage Management**:

To make the game Draw It or Lose It efficient in speed, an appropriate storage management system is needed. The best algorithm to request data from the hard drive is the shortest-seek-time-first(SSTF). This method searches for a request with the shortest seek time from the disk drive head. As a result, it significantly reduces the time to extract data from the hard drive. Another recommendation for the hard drive is the disk size. The storage size needs to be able to handle 1.6 gigabytes of picture data, game size, and user profiles. With how cheap storage is today, 500 gigabytes or a terabyte should be fine to handle such an application (Silberschatz, Galvin, & Gagne, 2009).

1. **Memory Management**:

The right memory management method is crucial for having a functional game application. A recommended method is using the swapping technique with a backing store on the hard drive. What happens, is that when the application needs the next process, it calls the process from the backing store. This is done by a CPU scheduler that uses a dispatcher to extract a process and swap it out with an unused process in the memory. Using this method helps prevent overburdening the memory and allows for maximum possible speed for the game Draw It or Lose It (Silberschatz, Galvin, & Gagne, 2009).

1. **Distributed Systems and Networks**:

The main goal of distributed software is having the ability to communicate between various clients with the server. One major factor to maintain compatibility and connectivity, is to use one programming language that is the backbone for the server network. Attempting to use different languages to talk to each other is going to increase the risk of a communication error and create a network outage. Java is the most used language and is available on all client operating systems. Another critical factor is utilizing a authentication method to allow only approved users to access the server for a game session. This helps maintain security and prevent overburdening the server with unnecessary connections (Silberschatz, Galvin, & Gagne, 2009).

1. **Security**:

Security is a must for any operating platform to prevent sensitive information from being compromised. A password is the most critical component for a user profile on the game application. It is recommended that passwords should be of significant length, various types of alphanumerical characters, and changed over a certain period of time. On the server platform, enabling group policy settings can add extra security to how the operating platform is used. When users access the server, group policy can block certain features and to allow only administrators to access them. Applying these two features will add significant amount of security for the game Draw It or Lose It (Silberschatz, Galvin, & Gagne, 2009).

References:

Apple Inc. (n.d.). MacOS Server. Retrieved October 05, 2020, from

<https://www.apple.com/macos/server/>

Eclipse Foundation, I. (n.d.). The Community for Open Innovation and Collaboration: The

Eclipse Foundation. Retrieved October 05, 2020, from <https://www.eclipse.org/>

Google. (2020). Android Studio. Retrieved October 18, 2020, from

https://developer.android.com/studio

Gewirtz, D. (2019, April 08). Windows, Mac, or Linux? We compare the pros and cons of these

computing platforms. Retrieved October 05, 2020, from https://www.zdnet.com/ article/windows-mac-or-linux-we-compare-the-pros-and-cons-of-these-computing -platforms/

Ionos.com. (2020). Linux vs. Windows: A comparison of the best web server solutions.

Retrieved October 05, 2020, from <https://www.ionos.com/digitalguide/server/know->how/linux-vs-windows-the-big-server-check/

Ismail, A. (2020, September 30). IPhone vs. Android: Which is better for you? Retrieved

October05, 2020, from <https://www.tomsguide.com/face-off/iphone-vs-android>

JetBrains s.r.o. (n.d.). AppCode: Smart Swift/Objective-C IDE for iOS &amp; macOS

Development. Retrieved October 05, 2020, from https://www.jetbrains.com/objc/

Lavieri, E. D. (2019). Hands-on design patterns with Java: Learn design patterns that enable the

building of large-scale software architectures. Birmingham: Packt Publishing.

Microsoft. (2020, October 02). Visual Studio IDE, Code Editor, Azure DevOps, &amp; App

Center. Retrieved October 05, 2020, from <https://visualstudio.microsoft.com/>

Silberschatz, A., Galvin, P. B., & Gagne, G. (2009). Operating System Concepts (8th ed.).

Hoboken, NJ: John Wiley &amp; Sons. Retrieved October 1, 2020, from https://learning.oreilly.com/library/view/operating-system-concepts/9780470128725/silb\_ 9780470128725\_oeb\_cop\_r1.html

SNHU. (2020). Module One. Retrieved October 04, 2020, from

https://learn.snhu.edu/d2l/le/content/524228/viewContent/9971677/View