

Draw It or Lose It Game Service

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

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

[**Table of Contents**](#_30j0zll)2

[**Document Revision History**](#_3znysh7)2

[**Executive Summary**](#_2et92p0)3

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

[**System Architecture View**](#_3dy6vkm)3

[**Domain Model**](#_1t3h5sf)3

[**Evaluation**](#_4d34og8)3

[**Recommendations**](#_17dp8vu)5

## [Document Revision History](#_3znysh7)

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| 1.0 | 09/19/20 | Josh Ryther | Created software design |

**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](#_2et92p0)

The Gaming Room’s most recent game, Draw It or Lose It, is an application currently only available on Android. They would like to develop a web-based version that can be ran on multiple platforms. The format of the service would be a single instance is created at start up. Games, players, and teams are created as they join the service and validated to avoid duplication or errors. This will ensure the user is given a consistent experience among all platforms.

## [Design Constraints](#_tyjcwt)

1. The application will need to be programmed using Java. The engineering team will need to be trained with this language for development. This is considered a constraint due to not all the engineers may have experience using Java. This is primarily a technical constraint but could impact budget and will require licensing of an Oracle commercial JRE account.
2. The application will need to be hosted on a cloud environment. Constraints can come from server capacity, data caps, speed throttling, and additional fees associated with traffic or other charges. This is a technical and business constraint. The technical limitations of the client’s current host will need to be evaluated or used to choose a new one. The choice of host may also affect the budget if the current application accrues more fees than the client expected.
3. The game and interface applications will be developed by another team. The game service application must be designed to interface easily and be well documented for use by outside vendors. This is a technical constraint since CTS is not being responsible for development of the game or interface applications.

## [System Architecture View](#_3dy6vkm)

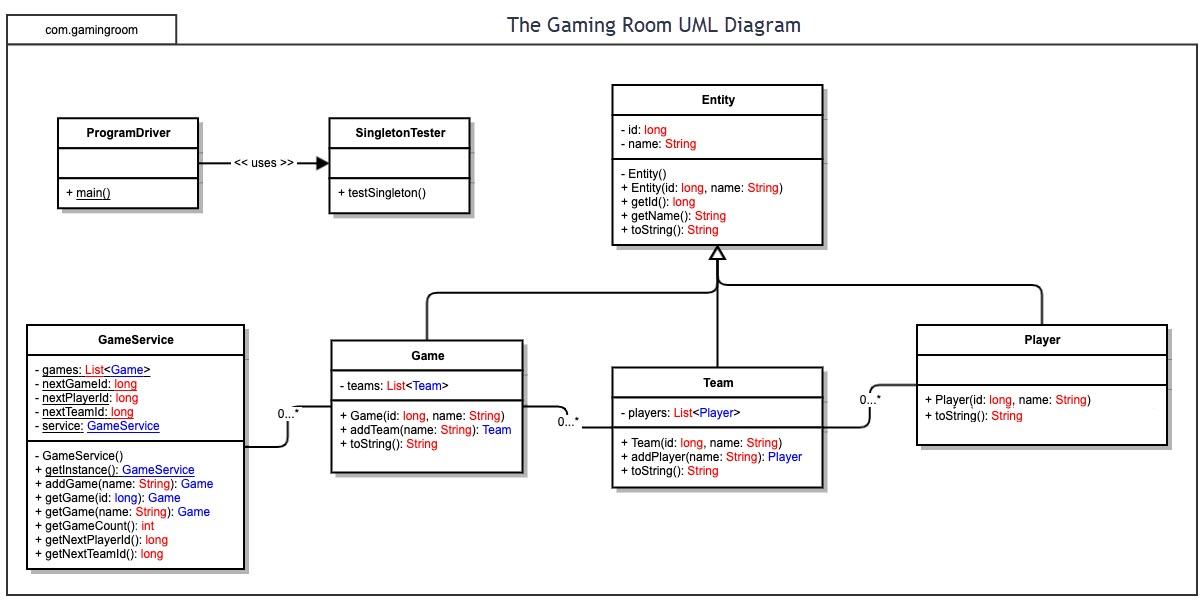
Please note: There is nothing required here for these projects, but this section serves as a reminder that describing the system and subsystem architecture present in the application, including physical components or tiers, may be required for other projects. A logical topology of the communication and storage aspects is also necessary to understand the overall architecture and should be provided.

## [Domain Model](#_1t3h5sf)

The Entity class is a parent class for the Game, Team, and Player classes. Id and Name are variables used in all three sub-classes along with their respective accessor methods. Each of the four classes also has a method that produces a string of the game Id and name. The Game class has an addTeam() method and the Team class has an addPlayer() method. Each method adds the respective team or player to their class’s array list if validation confirms it is not a duplicate. Validation occurs by using an iterator template to run through the array list and verify there is not a name match.

The Game Service class acts as the primary class for managing the online game lobbies. It uses a Singleton design pattern to ensure only one instance of the class is initiated. This is done by creating an object in the class then making the constructor private. Then a method is created that returns the object that was created for use in the main method as needed. Additionally, there are also various accessor and mutator methods as well for adding games to the service. An add game method uses an iterator pattern for validation and then adds the game object to Game Services array list if a duplicate match is not found. There are two methods that can search the array list and return a game by the name or index. The size of the array can also be accessed using the getGameCount() method.

The main method creates a variable for the single object of Game Service that can be created and invokes it using the getInstance() method. This is then used to add two games to the array list in Game Service. The Singleton Tester class is then used to validate that only one instance of Game Service can be created. It also creates a Game Service type variable and invokes the getInstance() method. It then prints the array list so the user can verify that the results are the same as what was already printed previously in the main method.



## [Evaluation](#_4d34og8)

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** | MacOS is a free OS but is only available on apple hardware. MacOS Server has recently lost a lot of functionality and now focuses on being a home and small business server that manages devices and media. Dynamic host configuration protocol and domain name services among many other features have been removed. A lot of manual configuration along with third party solutions will need to be utilized in order to use a MacOS server for an enterprise platform. | Linux is an open source system which means that the software is free. The company only needs to pay for the technical support required to install and maintain the server. Linux is also generally more secure, reliable, and easier to modify due to it's open source nature. | Windows will cost more than linux but will have better support through Microsoft. Integration with other Microsoft products is also superior than having to use more third party options. Out of the box Windows has more features built in. Linux servers are command line based while Windows has a GUI. | There are solutions to turn an android device into a simple web server. None of the solutions are meant for enterprise platforms. The intended use seems to be mostly for developers who want to test a website. The hardware is also not powerful enough to handle many users that enterprise platforms demand. |
| **Client Side** | MacOS accounts for only 10% of PC users. Developers must own a Mac and be familiar with Swift or Objective-C in order to develop for MacOS. This is a high cost for a low pay off in terms of the number of users. MacOS is based off of UNIX so any developer familiar with linux should also be able use the command line in MacOS. However, if the development team chooses to only deliver the application through pre-installed web browsers, the application should for the most part be platform agnostic and only run on HTML, CSS, and javascript. | Linux is designed to make development easier. The OS is free which helps decrease costs. It has support for most programming languages right away. The source code is available for modification if needed. However, linux does have a large number of distros. Sometimes an application may break on one distro but not others. This could greatly impact the amount of development and maintenance time needed. However, if the development team chooses to only deliver the application through pre-installed web browsers, the application should for the most part be platform agnostic and only run on HTML, CSS, and javascript. | Windows supports most programming languages but may need to be downloaded and installed before use. The OS is the most widely used consumer OS which means that development cost is worth the number of users that will result from it. Most developers should also already be familiar with how to use Windows due to its high adoption rate. However, if the development team chooses to only deliver the application through pre-installed web browsers, the application should for the most part be platform agnostic and only run on HTML, CSS, and javascript. | The engineering team will need to know how to develop for mobile devices. iOS follows a lot of the same limitations as macOS in that the developers will need to know how to program in either objective-c or swift. For android the programming language is primarily Java with an added benefit if they know kotlin. Both mobile devices have a high user base which would make both viable to develop for in relation to costs and benefits. However, if the development team chooses to only deliver the application through pre-installed web browsers, the application should for the most part be platform agnostic and only run on HTML, CSS, and javascript. |
| **Development Tools** | macOS and iOS programs can only be programmed on macOS so the apple developers will need to be provided hardware that can run the OS. Xcode is a popular IDE for iOS and macOS development. The company will also need an apple developer account in order to submit an application to the apple app store. | Linux has support for almost all languages. Developers can use a terminal or an IDE depending on what they prefer. Linux can be run on most hardware from high end PCs to raspberry pi’s so mid range PC’s should be sufficient to develop on. The development team will need to have the correct distros that they are developing for. | Windows can run on a large range of PCs. It is more demanding than linux so the hardware may need to be more powerful to run and compile faster. Visual studio code is a popular IDE for the OS but eclipse is also used for Java. Windows is a paid OS so each developer will need a license to run it. | iOS requires a mac in order to develop and submit to the apple app store. Xcode is generally the preferred IDE and will need to be installed. For android, Java must be installed or Kotlin depending on the requirements and knowledge of the development team. Most android applications are developed in android studio. |

**Sources:**

1. Gewirtz, D. (2019, April 8). *Windows, Mac, or Linux? We compare the pros and cons of these computing platforms*. https://www.zdnet.com/article/windows-mac-or-linux-we-compare-the-pros-and-cons-of-these-computing-platforms/.
2. Isberto, M. (2019, September 24). *The Best Operating Systems for Business and Personal Servers*. Colocation America. https://www.colocationamerica.com/blog/best-operating-systems-for-business-and-personal-use.
3. King, B. (2017, May 4). *7 Superb Reasons Why You Should Use Linux For Programming*. MakeUseOf. https://www.makeuseof.com/tag/6-superb-reasons-use-linux-programming/.
4. Vaughan-Nichols, S. J. (2018, January 29). ​The death of macOS Server. https://www.zdnet.com/article/the-death-of-macos-server/.
5. *Linux vs. Microsoft Windows Servers: The Official Volico Blog*. Miami and Broward Colocation | Volico Data Centers. (2017, April 17). https://www.volico.com/linux-vs-microsoft-windows-servers/.
6. Keizer, G. (2018, March 13). *Macs now account for 10% of active personal computers*. Computerworld. https://www.computerworld.com/article/3262974/macs-now-account-for-10-of-active-personal-computers.html.
7. Sneddon, J. (2018, January 29). *Is Native Linux App Development In Decline?* OMG! Ubuntu! https://www.omgubuntu.co.uk/2018/01/new-native-linux-apps.

## Recommendations

Analyze the characteristics of and techniques specific to various systems architectures and make a recommendation to The Gaming Room. Specifically, address the following:

**Operating Platform**: After careful analysis of the different available platforms I recommend to use Windows Server.

**Operating Systems Architectures**: Windows is a mature premium platform that has been used by businesses for many years. The reason for this is an unrivaled amount of support and features (2). Microsoft has many supplemental services that are well integrated into the ecosystem. By creating a Windows server it will also allow easy integration into Azure which has a host of features that allow scalability of the game. Among them is a serverless platform. By allowing Microsoft to manage the servers the client will not have to hire a team to implement and manage internal servers.

**Storage Management**: There are many data storage methods but the three that were considered for this project were contiguous, linked, and indexed. Contiguous storage stores blocks of data in a linear ordering on the disk. Linked allocation stores the data where there is room and uses pointers to connect the blocks of data that comprise a file. Indexed is like linked allocation but stores the pointers together in one location called an index block. Since the platform needs to be able to add or remove images from the game as needed the storage method needs to be able to quickly and efficiently be able delete and replace data. Contiguous allocation is not efficient since it has to move data to make room as it removes and adds to the disk. Linked is more efficient than contiguous but still requires iteration through the disks data to find what the program is looking for. Indexed allocation has the advantages of linked allocation but is arranged in a way to help the software access the data needed faster. Due to its efficiency in the context of the game indexed allocation is the best storage management method to use.

**Memory Management**: Memory management is the practice of using the available memory resources efficiently. Sequential and direct access are two methods of accessing data and loading it into memory. Sequential goes through the data in the order that the records are stored while direct accesses specific data locations. Since the game needs to be able to access and load specific images for each game session direct access is the most efficient method to use. Depending on the file sizes and memory limitations the client may also need to employ indexing to break up the files. Lastly, caching should be used on data that is often accessed to reduce duplicate resource requests.

**Distributed Systems and Networks**: Implementing a distributed system allows for one server system to be deployed and interact among many different clients. The clients can be programmed for their respective device and use HTTP requests over the internet to interact with the server. This makes the platform extremely scalable. Due to connectivity requirements the program will not be able to run offline. The user must maintain an internet connection to connect with the server and get the required data in order for the program to function properly. Internet outages or server downtime may therefore affect the service.

**Security**: In order to protect a platform, measures must be implemented at four levels. First the physical location of the server must be secure. Security personnel and authorization of employees must ensure only entry to those whose jobs require it. This will prevent unauthorized individuals from gaining access to the hardware. Second, access within the software must be layered to prevent unneeded permissions. This will help protect against accidental and malicious use of the system. Users are also susceptible to phishing attacks that can give unauthorized access to accounts. Training and password protections protocols must be implemented to minimize this risk. Third, the system must be created or set up to protect itself from breaches. Poorly written code can lead to an accidental DOS attack by the system on itself. Lastly, network security should be maintained. Encryption protects data from being intercepted and used without permission. Creating safeguards against potential DDOS attacks is also important for online services. A weak link in any of the four layers mentioned could create a system vulnerability. The more vulnerable the system is the higher the risk for a breach that could potentially be extremely harmful to the system or company reputation.

Source:

1. Silberschatz, A., Galvin, P. B., & Gagne, G. (2010). Operating system concepts. Wiley.
2. Linux vs. Microsoft Windows Servers: The Official Volico Blog. Miami and Broward Colocation | Volico Data Centers. (2017, April 17). https://www.volico.com/linux-vs-microsoft-windows-servers/.