

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.0

## 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 | 11/13/2020 | Matthew Neale | Initial draft |
| 1.1 | 11/14/2020 | Matthew Neale | Draft #2 |
| 1.2 | 11/15/2020 | Matthew Neale | Draft #3 |
| 1.3 | 11/28/2020 | Matthew Neale | Draft #4 |
| 1.4 | 12/18/2020 | Matthew Neale | Current draft |

**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 goal is to create a web-based version of the app Draw It or Lose It for the client The Gaming Room. The current app is developed for Android smart devices only. The idea of the game is to have a stock image render while players try to guess the image. The Gaming Room needs the web-based program to be able to run on multiple platforms. Java would be the recommended language to use as it can be run across multiple platforms. Requirements are as follows:

* The game must be able to support one or more teams.
* Each team must be able to support one or more players.
* Game and Team names must be unique, no teams with the same name.
* Only one instance of the game will be allowed to exist in memory at a time.
* Must be able to run on multiple platforms.

## [Design Constraints](#_2et92p0)

* The user must have a general understanding of how to use a computer and access a web page.
* The user must have internet access.
* The user must understand the concept of a game.
* The program will be implemented using Java.
* The game will be designed to work on any operating system.
* The program will use class inheritance and be able to be modified for future 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)

There is a superclass called Entity of which the classes Game, Team, and Player are subclasses. Game, Team, and Player all inherit from Entity. The class GameService is a singleton class meaning there can only be one instance of GameService in memory at any time. GameService creates an empty list of games. This list can be populated using the addGame(name) function. The list can include zero or more games. GameService also includes variables for next Game ID, Team ID, and Player ID. GameService also includes getter functions to retrieve the various names and IDs for Games, Teams, and Players. The Game class creates an empty list of Teams. This list can be populated using the addTeam(name) function. The list can include zero or more Teams. The Team class creates an empty list of Players. This list can be populated using the addPlayer(name) function. The list can include zero or more Players. The Player class holds the info for the player including ID# and name. All classes also include a toString() method to print to screen. The main function is included in the ProgramDriver class which uses the SingletonTester class to test for singleton function.

****

## [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** | **Pros:** Good if you are already running Mac OS software or if you are more familiar with Mac software. Unix based OS. Open-source servers. High security. User and file access controls. Support LDAP and ADP  **Cons:** Hard to find and more expensive. High cost for licensing fees. Can be done just as easily with Linux or Windows. Mac for hosting is not highly recommended. Docker support virtual only. Difficult to implement server-only environment | **Pros:** Free and open source. High security features. Cheap and easy to run. Most common operating system running on servers. UNIX based OS. User and file access controls. Support LDAP and ADP. Docker Support. The internet runs on Linux.  **Cons:** Linux is not widely used as an operating system on personal computers. Linux is constantly evolving, which means support may or may not be available for the version you are using. Need to decide on vendor support; Redhat or Ubuntu. | **Pros:** Not free but cheaper than Mac. Able to use various Windows features. More common and easy to find. Plenty of support for Windows friendly and Microsoft based technologies. High security. User and file access controls. Support ADP. **Cons:** Licensing fees. Not as secure as Mac or Linux. Windows is widely used in companies and personal computers, making it a bigger target for security threats. Need to use Server version. Limited LDAP. Poor open source server support | I have never heard of a mobile server host and it does not seem very practical. Servers require a large amount of memory and data storage that is not conducive to mobile devices.  **Pros:** iOS developer tools. iOS multi-device support (iPhone, iPad, watch, TV). Android developer tools  **Cons:** Android poor multi-device support |
| **Client Side** | Mac is very exclusive to Mac, which can be a problem dealing with other platforms. Mac can be more expensive than Linux or Windows. A client running MacOS would still be able to use an application hosted on Linux or Windows.  **Pros:** Support Safari browser, Firefox, and Chrome.  **Cons:** No IE. Small install of client desktops | Linux seems very popular among developers. It is free and open source; however, it is not commonly used on client side. A client running Linux still has compatibility with common files such as HTML, CSS, JavaScript, etc.  **Pros:** Support Firefox and Chrome  **Cons:** No IE. No Safari. Small install of client desktops | Windows is widely used and versatile. More expensive than Linux but more cost effective than Mac. Many people are familiar with Windows. Windows is compatible with all common web-based file types.  **Pros:** Support Large installed base of desktop clients. Firefox, Chrome and IE  **Cons:** No Safari | The client company already has a developed Android app. It would require more money, time and expertise to translate this to iOS. It would not be impossible to create an iOS version app if the prospective profit would make it cost effective.  + iOS: Firefox, Chrome, Safari + Android: Firefox, Chrome |
| **Development Tools** | Eclipse, Swift, HTML, CSS, JavaScript, Python, Ruby, Java. Multiple development teams specializing in server side and client side would be helpful.  **Pros:** iOS native and Android development supported. High-productivity environment. Large language support including Swift and objective-C + Java support + Unix shell scripting  **Cons:** Small pool of developers. Difficult /impossible to test IE browsers. Few IDEs, but high quality. | Visual studio, Eclipse, HTML, CSS, JavaScript, C++, Python, Java. Multiple development teams specializing in server side and client side would be helpful.  **Pros:** Large developer pool. New technology first platform. Largest set of IDEs with quality but not always support. Large language support including Swift and objective-C + .net support + Java support + Unix shell scripting + Powershell.  **Cons:** Android dev support only: native | Visual Studio, Eclipse, HTML, CSS, JavaScript, Java, C++, Python. Multiple development teams specializing in server side and client side would be helpful.  **Pros:** Large set of IDEs with support and quality. Large language support including C# + .net support + Powershell + Java support  **Cons:** Android dev support only: native | Visual Studio, Eclipse, HTML, CSS, JavaScript, Java, Python, Swift, C++. Multiple development teams specializing in server side and client side would be helpful.  **Cons:** Development is not common on mobile platform. |

## 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**: Linux would be an appropriate operating system to host a web-based application. It is used in most hosting servers.
2. **Operating Systems Architectures**: Linux uses the Linux kernel which is based on Unix. Linux is free and open source and highly adaptable to suit specific needs.
3. **Storage Management**: Linux uses a block storage system on block devices such as HDD or SDD. This storage can be expanded and partitioned as needed.
4. **Memory Management**: Linux divides memory into pages and refers to memory by page.
5. **Distributed Systems and Networks**: Linux is highly capable of running distributed systems and networks. Most servers currently in use are running Linux.
6. **Security**: Security in Linux comes from the fact that it is open source. Code can be reviewed by hundreds or thousands of people to discover bugs and how to fix them. Linux was created with security as a focus.

References

Horne, K. (2020, July 3). *Operating Systems And Web Hosting: Read Here Before Making Your Choices*. Who Is Hosting This. <https://www.whoishostingthis.com/compare/operating-systems/>

Silberschatz, A., Gagne, G., & Galvin, P. B. (2020). *Operating System Concepts 8th Edition* (8th ed.) [E-book]. Wiley. https://learning.oreilly.com/library/view/operating-system-concepts/9780470128725/silb\_9780470128725\_oeb\_c14\_r1.html