

The Gaming Room

# **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 | 02/17/2021 | Madeline Neel | Review characteristics, advantages, and weaknesses of Linux, Mac, and Windows operating system and mobile devices and give recommendations based on the findings. |

**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)

Draw It or Lose it is a web-based application, which is similar to the television game *Win, Lose or Draw*, that will use images from a library of stock drawings as clues. The images will slowly reveal itself and be complete within 30 seconds. If that team fails to uncover the puzzle in the given time, the remaining teams have 15 seconds for their attempt at solving the puzzle. Draw It or Lose it will consist of four one minute rounds.

## [Design Constraints](#_2et92p0)

- Draw It or Lose it needs to be able to run on multiple different platforms.

- This will differ between platforms due to different screen sizes, servers weaknesses and strengths, and different programming languages.

- One instance of the game can exist in the memory at a time.

- Creating unique identifiers for each instance (game, team, player) will allow this to be accomplished.

- A game will allow one or more teams involved.

- Each team will have multiple players assigned to it.

- Game and team names must check if name is already in use.

These design constraints follow the software requirements that the client has requested. Since this application will be run on multiple platforms, it may require more developers to complete each programming language required for the different platforms. This may, also, cause variations in the design template of the game.

## [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)

To begin with, the ProgramDriver has the main() method, which is used as the entry point into the application. ProgramDriver then uses the SingletonTester to test the mains singleton. Game, Team, and Player all inherit from Entity. GameService has zero to many Game(s), Game(s) have zero to many Team(s), and Team(s) have zero to many Player(s). In simpler terms and also from the software requirements, it shows that GameService can have multiple Game(s), each Game can have multiple Team(s), and each Team can have multiple Player(s).

****

## [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** | Advantages:  MacOs is flexible to run almost any operating system using virtualization, virtual machines, and/or dual boot. Secure and provides the best IDE available.  Disadvantages:  MacOs has limited hardware options. Entry level Mac devices have limited memory storage. | Advantages:  Commercial, free, and/or mostly open source software available. Easy encryption options. Anti-virus software is not needed. Command prompt is advanced.  Disadvantages:  Not a completely open source; can’t entirely modify operating system. Restrictions limit ability to run virtual machines. Hardware drivers are not available. | Advantages:  Hardware can be changed easily. More software available.  Disadvantages:  Not as secure. Can’t use Linux command line. | Advantages:  Connectivity- cellular networks.  Disadvantages:  Visual element will need to be evaluated to fit screen size – touch screen. Some mobile devices have less storage and processing power. |
| **Client Side** | Medium user base. Profitable app store, but fewer games are released on app store. Forces users to purchase Apple Macs. Costly. Around 3-9 months to develop. | Smaller user base. Cost friendly. Around 3-9 months to develop. | Larger user base. Costly. Around 3-9 months to develop. | Since easily accessible, larger user base. Costly. Around 3-9 months to develop. |
| **Development Tools** | Swift, AppleScript, C, Objective-C, C++, Ruby, Python, PHP, zsh, bash, x86, ARM assembler | Has integration with iOS.  Python, C++ Python, C, Perl, Java, Shell | C#, Windows API | JavaScript, Kotlin, C++, C#, Python, Swift, Objective-C, PHP |

## 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**:

I would recommend using Windows operating platform. Specifically using Java as it is a cross platform language that is available on mostly all platforms. On Windows, Eclipse is available to develop Java desktop applications. For Mac, you can use the tool Jar Bundler to turn jar files into Mac applications. Linux, already, uses Java to create applications. Mobile devices use Android Studio and Java to create Android apps. Another possible programming environment that could be used is Xojo. Xojo’s cross-platform development tools allow mostly any code to be used for any platform.

1. **Operating Systems Architectures**:

Windows is a layered design, meaning it has two main components – user and kernel mode. Kernel mode allows the executing code to have unrestricted access to the hardware while user mode limits the access. Windows has both 32-bit and 64-bit versions. It is, also, a modular structure – the modules include hardware abstraction layer, kernel, microkernel, executive services, environment subsystem, and integral subsystem.

1. **Storage Management**:

Storage management will allow the game to store and access long-term data. The images of drawings given to the user will be from a large library of stock drawings that are held in storage. In order to determine how much storage we need, we would need to calculate how many drawings will need to be held and any additional information the client may want to keep available. Other additional information could include, the users win-to-loss ratio from previous games being provided on the users profile and more. A type of storage that could be used is a cloud database. This can then access the storage by using REST API, which will use HTTP requests to access and use data.

1. **Memory Management**:

Memory management holds short-term data and will be necessary to have a faster processing time than storage. As the memory will most likely result from the users devices, it is important to give the memory capacity and other requirements to the user before downloading the application. If not, the application may not run rapidly and effectively; which may affect the users thoughts on the game. In order to determine the exact amount of memory needed, we would need to calculate how many users can join a game and how many guesses per person will be provided throughout that game. These guesses will not be held in storage, but will need to be held in memory for the remainder of the given game to determine a winner.

1. **Distributed Systems and Networks**:

Distributed systems allow availability, performance, reliability, scalability, manageability, and cost. This will accomplish having a network that connects the devices though an IP address.

1. **Security**:

Mac and Linux have the best security but Windows Security is available to protect your device and information. Using firewalls and a VPN will help protect information between various platforms. Another reasonable precaution would consist of a third-party security audit to test the apps code.