

Draw it or Lose it

# **CS 230 Project Software Design Template**

Version 3.0

## 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 | 01/18/21 | Travis Garlick | -Added summary of design constraints, problems, and solutions  -Added description of the UML class diagram |
| Version | Date | Author | Comments |
| 2.0 | 02/01/21 | Travis Garlick | -Edited summary and constraints  -Added comparison of development between platforms |
| Version | Date | Author | Comments |
| 3.0 | 02/15/21 | Travis Garlick | -Completed my recommendations to The Gaming Room |

**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 game application made by The Gaming Room in which players must solve a puzzle within a time limit. Our goal is to take this android based game, and design versions of it for new platforms. This will first require researching the characteristics of each potential new platform, so that we can compare them to determine which ones we want to develop this application for.

## [Design Constraints](#_2et92p0)

This is an application intended to be distributed over the internet, so we must try to make sure it can be accessed and run by any device and OS that can use the web.

Having decided to make this web-based game, there are a couple of decisions that need to be made before the application will be ready to distribute.

1. Choosing the platform to develop the application on.
2. Assigning a programmer or team the time to develop the application.
3. Testing the application on any platform it is expected to be used on.

## [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)

On the following page is a UML Class Diagram that models the structure of this application. It details 7 classes within the Gaming Room package. There is one abstract parent class (Entity) with 3 child classes (Game, Team, and Player). Each of the three child classes inherits both private attributes of Entity, and all 5 of Entity’s methods (both private and public). The Game and Team classes each have an additional unique private attribute, and all three child classes have new unique public methods. The GameService class, associated with the Game class, has 13 various unique attributes and methods. There can be any number of each of these four classes at the bottom of the diagram. The upper left of the diagram shows the driver class containing the main method that runs the program, and the SingletonTester class it uses to check for duplicate name entries.

****

## [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** | + High flexibility for changes to server or accessibility. | + flexible  + versatile  + lowest cost. | + Most commonly supported as a server platform. | - Lowest capacity for hosting resource heavy servers. |
| **Client Side** | + More secure than Windows.  - Less users than Windows or Mobile. | Current version of the game is already made on Android, which is just a different version of this OS. | + Largest user base.  -Most targeted platform for malicious entities. | - Mobile browser is cumbersome, a dedicated app might be necessary, or at least a separate “mobile version” of website to accommodate screen size. |
| **Development Tools** | Programming language: Swift  Development tool: Xcode | Most dev tools, like Eclipse IDE, and most programing languages have useable versions on all platforms now. | Primary platform of many tools, like Visual Studio, and Notepad++ | (since this app is already on Android, I will focus specifically on IOS here)  iPhone shares dev tools like Swift and Xcode with Mac. |

## 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 weighing the pros and cons of each platform, I would recommend that any physical server requirements that The Gaming Room has for this application be filled by a Linux system. Linux servers would provide all the resources needed of a server at the lowest cost.
2. **Operating Systems Architectures**: Linux is a popular variety of the UNIX operating system based around the monolithic Linux kernel.
3. **Storage Management**: Linux stores data in either 8 KB blocks or 1 KB divisions, accessed through the page cache.
4. **Memory Management**: In Linux, physical memory is separated into three different zones that separate the amount of memory below, between, and greater than 16MB to 869MB. Linux’s page allocator uses the buddy system which frees memory in pairs of increasing size.
5. **Distributed Systems and Networks**: Since this is a web-based application, the choice of host system should not affect the ability to distribute the game to a variety of different user systems.
6. **Security**: The first level of security will of course be the client with which the user interacts. The game app will have a unique player name for the user to log in to the game with. This game does not seem to be recording any sensitive user information (like financial data, or the user’s location), so I do not suspect security at this level will be a large concern.

The next level of security will be the communication between the client and the server. Still no sensitive information should be involved here but if this communication were blocked by another party, or a denial of service were to be directed at the server, then it could interfere with the player’s ability to use the application and potentially lose The Gaming Room some users. This is the one level that will implement Linux’s specific system authentication.

The final level of security is the code that runs the application. If an exploit exists in the program (like an input buffer overflow) that allows protected access to be violated, it would leave the user’s entire operating system vulnerable. This would be the most disastrous security failure possible.