

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 | 05/18/21 | Paige Sandler | Summary, constraints, and domain model added |
| 1.1 | 06/03/21 | Paige Sandler | Development Requirements table |
| 1.2 | 06/19/21 | Paige Sandler | 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 Gaming Room wants a web-based game to run on multiple platforms which is based on their current android app, Draw It or Lose it. Each game takes place over four rounds lasting one minute each. The application “draws” an image from a library of stock images. The image completes at the 30 second mark, and if the team does not guess the puzzle before the minute is up, it gets passed around to the other teams who each get one guess with a 15 second time limit.

The game must allow one or more teams to exist and play. Each team should be able to have multiple players join. Both team and game names will be unique, and only one instance of the game can exist at a time. Each player, team, and game will have unique identifiers so the app can check during creation. An iterator can be used to check for teams and players, and a singleton pattern can be used to verify that only one game exists at a time.

## [Design Constraints](#_2et92p0)

The application will need to be tested for cross-browser compatibility. Some browsers, such as Safari 12 and later, do not support the Java language. Hardware requirements are to be determined and will need to be examined later. To ensure a seamless experience across all operating platforms, development may need to be outsourced depending on the language used.

## [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)

The Game, Team, and Player classes all share common attributes which they inherit from the Entity class, a super class. This is an is-a relationship. The Player, Team, Game, and GameService classes are also related to each other via a Has-A relationship. The GameService class may have multiple Games, the Game class may have multiple Teams, the Team class may have multiple Players.

****

## [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** | Mac runs Apache servers which are similar to Linux. They tend to be on the more expensive side, but the hardware and software is known for being top of the line and very secure. Cost overall is going to depend on whether the company wants to host on their premises or pay a company to host for them. | Linux is much more cost effective than Mac or Windows in this category. It requires less resources to run, and is open source. Linode and Digital Ocean would be cheaper solutions than say, AWS or Firebase. | Windows servers can be a bit slower than the other two, but has a bigger variety of software to choose from that can not be run on the other operating systems. | There is a large list of mBaaS providers to choose from. Many are pay as you go services and some have free self-host features. A flat fee package would make costs more predictable. They allow integration with social networking sites as well as other features like push notifications. |
| **Client Side** | Mac price tends to be higher. Many applications on Mac systems are written in Swift or Objective C. React Native can be used instead across all platforms to save time or money, but may not have the same performance or flexibility. Using different teams to create independent apps may be costlier and take considerably longer. | Linux cost is generally lower than other platforms, but also takes longer to learn than Mac or Windows. React Native would work here as well. | Windows has a lot of support for web development and is generally easier to work with. They are less costly and have more configuration options. Windows has also been starting to embrace open source technology. | There are multiple operating systems across different phones, making implementing a uniform application across all systems more complicated. Catering to the specific OS of the phone may require more time and higher costs, but would result in an application that runs more smoothly across platforms. Screen sizes should be taken into account (ex: a tablet is generally much larger than a phone), and the UI should be intuitive for touch controls. It is also worth noting that there is a yearly cost for a developer account on the iOS platform. |
| **Development Tools** | Xcode IDE, Git  HTML, CSS, JavaScript, React Native, Swift, SQL  A hybrid app rather than a native app could be more cost effective in the long run, but performance issues are a risk down the line with maintaining the app. | Eclipse or IntelliJ  Git  HTML, CSS, JavaScript, React, MySQL | Microsoft Visual Studio, Eclipse, Git for Windows  HTML, CSS, JavaScript, React, Java, MySQL | Android Studio, Xcode, Microsoft Visual Studio  HTML, CSS, React Native, Swift, Objective C, JavaScript  Could utilize different teams for the different systems here. |

## 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 recommended platform for this application is Linux. The reason behind this OS can be run on multiple architectures and systems, with no additional licensing fees.
2. **Operating Systems Architectures**: Linux is open source and community driven. It uses an Extended File System and a Unified File System for storage.
3. **Storage Management**: A solid state drive would be the recommendation here to produce faster load and run times than a traditional hard drive, especially as there are many images that the game will need to access.
4. **Memory Management**: The system creates a virtual memory layer and then assigns processes to it. This is both part of the RAM of the system and swap space from the hard drive (this is an extra pocket on the hard drive reserved as backup in case the system runs out of usable RAM). This also prevents processes from overlapping and attempting to use memory that is already being used by another process.
5. **Distributed Systems and Networks**: While the application appears to all be one interface to the user, it is spread out across multiple systems which communicate with each other and share information. This makes the application more stable, as one system failure doesn’t amount to the entire service going down. Workloads are broken up between machines, which is much more efficient than having one machine trying to do all the tasks. Adding additional nodes to the system is a simple inexpensive process if necessary, for expansion in the future. One thing to keep in mind would just be if the system is too spread out, that may cause latency in communications between systems.
6. **Security**: Linux web hosts have HTTPS support and native support for end-to-end encryption. The OS sees all processes as files and uses a traditional read/write/execute system to items stored in the OS. Permissions can be granted for access to specific areas of the application, such as game images. The root user on each Linux system should be disabled to keep potential threats from gaining access to the system, with a secondary user being created for administrative purposes. Each user should have their own permissions based on their role, and no end user should have access to admin privileges.