

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 | 09/19/20 | Nicole Mahoney | Requirements and Design Constraints |
| 1.1 | 10/1/20 | Nicole Mahoney | Operating System Comparison |
| 1.2 | 10/17/20 | Nicole Mahoney | 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 would like to set up a web-based environment for users to play their game, Draw It or Lose It. This game must be accessible on multiple platforms. Each game and multiplayer team will have a unique ids as identifiers.

## [Design Constraints](#_2et92p0)

1. Each game will need to have one or more teams playing, requiring all users to be shown the most updated version of the game in real time.
2. Each team will need to have multiple players assigned to it, allowing those users to be in the same game state at the same time.
3. Game and team names must be unique, so the system must be able to quickly check if a game or team name are already in use.
4. Only one instance of the game can exist in memory at any given time, so the software must have a way to verify the game instance and ensure a duplicate is not created.

## [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 ProgramDriver inherits attributes and methods from SingletonTester, so that it is able to run the singleton test in it’s main method. Game, Team, and Player inherit attributes and methods from Entity. Entity encapsulates id, name, and the Entity constructor by making them private so that a duplicate instance of the same entity cannot be created. Game encapsulates its Team list and Team encapsulates its Player list. GameService and Game are associated and do not depend on each other. Team, Game, and Player objects are polymorphic because they are also an Entity. Team, Game, and Player classes are also portable because they can exist on their own and could be used in other places in the software. The ability to addTeam and addPlayer are public so that it can be updated and changed by the user.

****

## [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** | MacOs uses the Unix OS kernel and open source servers. This makes MacOS a secure option as security vulnerabilities are more easily identified and corrected. MacOS also has user and file access controls, giving developers more control. | As Linux is open source, it is secure as security vulnerabilities can be addressed and corrected more regularly. A choice of vendor support exists. There is a large online and cloud support given that the internet runs largely on Linux. | Windows is secure, although not open source making security vulnerabilities harder to identify. There is a larger support for Windows as it is so widely used. | Multiple device support exists within Apple products such as iPad or iPhone, and also with Android. |
| **Client Side** | MacOS has its own exclusive Safari browser. It also supports Firefox and Chrome, which are supported by Linux, Microsoft, and mobile devices as well. This means wider support across more platforms which would be both cost efficient and time efficient for developers. | Linux is able to run both Firefox and Chrome, although not Safari and Microsoft Edge. This would only be useful cost-wise if only development on widespread browsers was desired. | Windows is the most popular user platform and therefore would reach the most users. Developing to include Firefox and Chrome as browsers also allows less used user platforms like MacOS and Linux to be included as well. | Both iOS and Android are able to run Firefox and Chrome, so as long as these popular browsers are included in development, they will be far reaching to smartphone users. This maximizes benefit from cost and development. |
| **Development Tools** | iOS and Android development is supported, allowing smartphone users to be reached despite not being on a MacOS platform. Arduino, an open source IDE can be used on Mac OS. Visual Studio Code can be used for Javascript. | Can support Android development but not iOS. There are a variety of IDEs but the availability of support is not as wide. BlueJay Java IDE, Scratch, and Geany are some of the many IDEs available. Supports HTML, CSS, Java, Javascript | Supports Android development but not iOS or MacOS. IDEs such as Eclipse, Visual Studio, PyCharm, can be used. There are a large variety of languages supported by Windows such as C#. |  |

## 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**: Windows is the recommended operating platform for this game. Windows is the most widely used platform, reaching the most users. It’s compatibility with Chrome, Firefox, and Android OS make it possible to expand to other platforms in the future. Although Windows is not compatible with MacOS and iOS development, the availability of Chrome and Firefox for MacOS and iOS allow those users to participate as well.
2. **Operating Systems Architectures**: Client Server architecture would be useful in this app.
3. **Storage Management**: A cached storage management system would be recommended as information needs to be quickly retrievable by clients.
4. **Memory Management**: The operating system will use multithreaded processes to create and run multiple instances of each game at the same time. These will run as separate processes allowing more than one game to be played at a time.
5. **Distributed Systems and Networks**: Distributed systems allow the look and feel of software running from one system when infact the workload is distributed amongst many different systems. If there are outages it would only affect one or some computers and not affect the whole network.
6. **Security**: Windows is a secure operating platform. As the game Draw It or Lose it does not require personal information or special permissions from the user, security vulnerabilities through Windows will have minimal effect. We encourage Draw It or Lose It to encourage users to use login information that they do not use for any other sites. It is suggested that email address is not used for log in therefore not potentially exposing pertinent user information.