

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 | 05/23/2021 | Chris Tapia | First iteration of game design and model |
| 2.0 | 06/08/2021 | Chris Tapia | Updated content under the Evaluation section |

**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 to expand their game from Android to the web and eventually to other operating systems (OS) while ensuring that their requirements are met. Games will include one or more teams while maintaining only one instance of each game at a time. It’s important to note that game and team names must be unique.

## [Design Constraints](#_2et92p0)

* Developing a game in a web-based environment requires the use of other languages and tools such as JavaScript and HTML
* Client wants a game based on the web, which will require specific hardware and an operating system to run on the web server
* Since this game requires unique identities for players, security will be a huge focus to ensure of that everything stays unique
* Game will be using a lot of images which can out a strain on storage space. Amount of required storage will be considered

## [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)

Below is the design model for the application. The Entity class is the concrete class that gets inherited to the Game, Team, and Player classes which hold the data for the Game. The Player class grabs the data for each player and communicates that to the Team class. The Team class grabs all the players and is assigned a name which gets checked if it’s unique or not. The Team and Player classes will grab the data for each team and send the info to the Game Class. The Game class is known as the pattern class which helps design each instance of the game by communicating it to the GameService object class. Essentially, the GameService class creates each team that is playing. This allows for having multiple teams playing at the same time with only one instance of the game existing at any time. The relationship between the pattern and the object class is known as a Singleton Class.

When viewing the diagram, you’ll also notice the 0..\* which identifies a zero to many relationship.

****

## [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 doesn’t have a lot of tools for development from the get-go which can allow for a slower start to development. But once all of the proper tools are downloaded, it can be fairly easy to get things done. As far as hosting web-based software, it can still be done because Mac has the same access popular internet browsers as other environments.  A server license through Mac OS X Server starts at around $29.99/mo | Linux has been known to have the least restrictions which can make development a breeze. But it’s important to note that there isn’t much support or use of Linux outside the development space. With that being said, Linux can still host a web-based application without any issues.  You can run a server on Linux using services like Ubuntu for no costs of a license fee at all. There are Enterprise options through Red Hat Enterprise at about $799/yr for a Standard subscription. | Windows is great for a wide-range of development tools and allows for a bit of freedom when hosting web-based software. When compared to Linux, Windows has more overall restriction.  A server license through Windows costs about $20/mo to lease or $972 to own for it’s Standard Edition. For its Datacenter Edition it can be about $125/mo to lease and $6,155 to own. | Applications on Mobile Devices would prove to be very resourceful, but they can take longer to develop. This is mainly due there being different operating systems (OS) and the restrictions that each OS has. |
| **Client Side** | With Mac offering a lot of tools for development, costs and timeframe should be considered about the same when comparing to Windows. We also have great expertise with this environment. | Due to its simplicity and native development tools, developing on Linux would be the fastest and most affordable solution. | Developing software in Windows would be fairly easy given the number of tools that are available. | With the need to cater to multiple OS, the process for development Mobile Devices can take longer than hosting on Windows or Mac. But the benefits of having your application on a Mobile Device is a great investment since it allows users to take your game on-the-go. |
| **Development Tools** | Java, Eclipse IDE, Terminal.  Eclipse is free and an open-source software. | Java, Eclipse IDE.  Eclipse is free and an open-source software.  Linux also has it’s own built-in command line which makes it easy for programming with. | Java, Eclipse IDE, cmd.exe.  Eclipse is free and an open-source software | Swift, Java, Android Studio IDE, Xcode ID.  Android Studio is free for commercial use but costs a one-time registration fee of $25to upload to the app store.  Xcode is also free to use but to join Apple’s developer program which includes uploading to the app store, it can cost as low as $99/yr depending on the membership. |

## 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**: Mac. This would allow for an easier transition to Mobile Devices.
2. **Operating Systems Architectures**: Mac has a lot of tools with lots of support, thus making it a good foundation for hosting the web-based application. Mac also has access to Android Studio and Xcode, whereas Windows does not have access to Xcode.
3. **Storage Management**: MySQL
4. **Memory Management**: Using Java on Mac is essentially the same as using it anywhere else. Java handles memory management very well with the use of its built-in garbage collector.
5. **Distributed Systems and Networks**: To make the game work across multiple platforms, this can be done using the broker pattern and having a middleman to coordinate the communications coming in from each OS through the network. In doing so would increase efficiency. This would make it difficult should a player using a mobile device, lose signal or someone were to experience a power outage. There would have to be a system in place where, if a loss of connection is detected, it would just remove that player from the team.
6. **Security**: To ensure high-level security, we will utilize the Iterator design pattern which allows access to an object’s members without sharing the encapsulated data structure. On top of that, Mac is regarded for its high-level of security, so you can have peace of mind that your game is secure.