

# **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 |
| --- | --- | --- | --- |
| 3.0 | <08/13/21> | <Alex Grimes> | Final Recommendation |

## [Executive Summary](#_sbfa50wo7nsh)

We are looking to make a web-based game Draw it or lose it this will be a game based off of an 80s tv show where our contestants will be presented with images for a pre-determined time and expected to make a guess based off the images they are presented if the player/team is unable to answer the opposing team should have the ability to make a guess as well we only want a single game instance at a time and all identifiers should be unique.

## [Design Constraints](#_2et92p0)

Design constraints for web-based things are not to high mainly we will want it to look good on any size screen or at least as many as we can feasibly give the time we have for development. The size of the game should be relatively small and should properly report errors as well. Should also limit images and resources to things that can be easily packaged with the application on the web server.

## [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)

Our Subclasses Game, Team and Player inherit from our superclass Entity the 0...\* on our subclasses mean that there will be anywhere from zero to many of those classes in use at any time during the program. There are also private methods represented by – signs and public methods represented by + symbols. If a method is private, it can only be called from the class, it is listed. Under whereas a public class can be called from anywhere in the class.

****

## [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** | Closed System have to wait on Apple to release updates fairly expensive | Open System with nearly infinite distros available that anyone can update much cheaper than windows and mac | Very similar to mac in terms of closed system rely on Microsoft for updates and will be more expensive than Linux | As far as I am aware there is no realistic solution to running a server from a mobile device to meet these needs |
| **Client Side** | As far as web based goes on the client side for mac this could be viewed in any number of browsers Safari being the mac specific browser but Firefox Chrome and even edge are possibilities to consider. | All OSs have access to most web browsers these days so dev time should be given to optimizing the application to run smoothly and work on various resolutions and screen size. | All OSs have access to most web browsers these days so dev time should be given to optimizing the application to run smoothly and work on various resolutions and screen size. | Mobile will be the one exception certain concessions may need to be made to make the app work on desktop browsers and mobile browsers this will likely eat into time and resources that could be used elsewhere if mobile development were left out |
| **Development Tools** | All tools and languages can be used on MacOS with some difficulty using languages like c# however with the Visual studio IDE you can develop easily in any language | Linux has some more limitations on IDE choice and development tools but most of them can be made to work with a bit of massaging however it will require more effort than just clicking install. | Windows is similar to mac as pretty much all languages and IDEs will work for it with the exception of Swift and XCode those are mac and IOS dependent can probably still be ran in a virtual environment with a bit of effort and time, however. | Would not recommend doing development on a mobile platform typically when developing for mobile you use a VM to simulate what the mobile OS you will be using is and run the program that way to test. |

## 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**:

By far the best Operating platform to go with if you are trying to expand to other environments is going to be Mac or Linux not only are mac and Linux compatible but the mobile environment is also based on a similar platform making porting the game extremely easy when compared to going from windows to a Linux stye environment.

1. **Operating Systems Architectures**: This expands on what I was stating above macOS is based on a Linux kernel whereas a windows environment has its own NT kernel design that is used on windows devices and Xbox exclusively. Although there certainly are ways to run a Linux kernel in windows so it can still be considered. Linux and Mac both offer an array of architectures to choose from as well Mac and Linux both offer x86 and ARM. Linux does have a leg up on updates though due to its Open-source nature.
2. **Storage Management**: We could go with a software like Suse that would allow us to store our data in a way that it could be accessed no matter what platform is used currently or in the future effectively helping future proof the setup. All solutions could also be configures with NAS and or direct storage solutions and combined with a raid 5 implementation can be very efficient and safe.
3. **Memory Management**: MacOS offers an always on virtual memory system providing up to 4Gb of addressable memory making it amazingly effective at this task. Both Linux and Mac support a huge array of database options a few of my Favorites for Mac have been Postgres, MongoDB, and MySQL.
4. **Distributed Systems and Networks**: Since this is going to be a web application then the internet would be a must assuming this will be ran on a webserver using a stack like MAMP The name *MAMP* is an [acronym](https://en.wikipedia.org/wiki/Acronym) that stems from the names of the components of the system:[[1]](https://en.wikipedia.org/wiki/MAMP#cite_note-1) [**macOS**](https://en.wikipedia.org/wiki/MacOS) (the [operating system](https://en.wikipedia.org/wiki/Operating_system)); [**Apache**](https://en.wikipedia.org/wiki/Apache_HTTP_Server) (the [web server](https://en.wikipedia.org/wiki/Web_server)); [**MySQL**](https://en.wikipedia.org/wiki/MySQL) or [**MariaDB**](https://en.wikipedia.org/wiki/MariaDB) (the [database management system](https://en.wikipedia.org/wiki/Database_management_system)); and [**PHP**](https://en.wikipedia.org/wiki/PHP), [**Perl**](https://en.wikipedia.org/wiki/Perl), or [**Python**](https://en.wikipedia.org/wiki/Python_(programming_language)) ([programming languages](https://en.wikipedia.org/wiki/Programming_language) used for [web development](https://en.wikipedia.org/wiki/Web_development)). Or LAMP if you wanted to replace Mac OS with Linux. REST APIs (Representational state transfer) is a standard used by Most web developers to ensure that Transmission like the pictures in our app can happen smoothly between the client and the server. We should also place a size limit on the Pictures to never be larger than our current largest file.
5. **Security**: The Mac OS recommendation is by far the most secure choice for the platform. The smaller user base means that it is a smaller target for a lot of these large collectives building malware it also has a very streamlined locked down OS that makes it difficult to accidentally allow malware into your system along with apple's very regular update system this would be an extraordinarily strong contender. There is also a wide array of third-party software options available for any platform but Linux and MacOS seem to have the most support and variability. There will also be the need to get certificates from a CA one of the main ones for a web app should be an SSL certificate.