

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)4

[**Recommendations**](#_m8aleynsvzvc)8

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 4.0 | 02/16/21 | Elizabeth Hodgman | Edited Evaluation & Finished Recommendations (Finished Software Design) |

**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 has contacted CTS to assist in developing an environment for a web-based version of their gaming app, “Draw it or Lose It”, that is currently only an Android App. They want help in streamlining the development to allow their game to serve multiple platforms. The application will need to render images from a large library of images. Multiple teams with multiple players can play in a game. More than one team cannot have the same name. Lastly, only one instance of any given game can exist in memory at any given time.

## [Design Constraints](#_2et92p0)

* Needs a large amount storage to hold the stock image library
* Needs a server that can handle having a lot of people using it at any given time without getting bogged down
* Would require a server administration system for management
* Will need a system, such as a username and password, to identify user’s level of authority (Admin, User, ect.)
* Different platforms will require using different programming languages and expertise
* Web-based game should perform similarly to the Android App
* If there is any time frame or budget that needs to be followed

## [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 class holds the main method that runs the software. ProgramDriver uses SingletonTester to make sure only one instance of GameService exists within memory. Entity carries attributes and methods that Game, Team, and Player inherit. One of these methods exhibits the polymorphism pattern. The toString() method is within the Entity, Game, Team and Player classes. This method returns the classes id and name. The Player class collects the game players name and id. Team class puts the player information in a list. Then, Player class gets the team name and id. Game class puts the team info into a list and gets the game name and id. Finally, GameService makes a list of games and takes the information to run one instance of the games at any given time.

****

## [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** | **Advantages:**  - Offers Server Deployment and is easy to set up  - Stable system  - Good security  **Weaknesses:**  - Licensing can be pricey  - Must use and develop on an Apple Computer  - Requires previous knowledge of the system  - May have to hire or train developers | **Advantages:**  - The cheapest option for licensing (Free)  - Stable system and high processing power  - Open source network making collaboration easy  - System has good security  **Weaknesses:**  - Requires more expertise using command line and managing configuration files  - Not compatible with some Windows applications | **Advantages:**  - Offers deployment service (WDS)  - Provides requirement for building with .NET languages  - Application and Access compatibility  **Weaknesses:**  - Licensing price can be expensive depending on what is needed  - Some licenses have Core requirements, CAL requirements, and additional licensing may be needed | **Advantages:**  - Inexpensive to run a server  - Easy deployment  - Allows for a portable web server.  **Weaknesses:**  - Requires a third-party app install to manage server  - Most likely would not be powerful, stable, or as secure  - May have to hire or train developers |
| **Client Side** | **Pros:**  - Safari has built-in integration and sharing on Apple products  - Safari is secure for Apple users  - Safari has access to saved passwords  - Firefox is open-source and more secure  **Cons:**  - Safari can have compatibility issues  - Flash does not work well on Safari  - Firefox and Chrome have high RAM consumption | **Pros:**  - Chrome offers same extensions to Linux as any other OS  - Both Firefox & Chrome are high-speed browsers  - Firefox is optimized for Linux  **Cons:**  **-** Can run Safari, but it is very difficult to install on Linux OS  - Firefox and Chrome have high RAM consumption | **Pros:**  - Firefox is open-source and more secure  - Chrome connects Android user information  - Both Firefox & Chrome are high-speed browsers  -Microsoft Edge takes up less RAM  **Cons:**  - Safari doesn’t run well and isn’t as secure  - Microsoft Edge no longer supports Flash player  - Firefox and Chrome have high RAM consumption | **Pros:**  - Multiple browsers to choose from  - Different devices usually have a main browser already installed that are more commonly used (Android: Samsung Internet Apple: Safari)  **Cons:**  - Layout needs to be tailored to display on a mobile device browser  - Some mobile browsers are less powerful |
| **Development Tools** | **Pros:**  - Xcode is a free IDE that has a wide variety of language packages  - Xcode includes Java, HTML, and Javascript  - Visual Studio for Mac has better compatibility with other OS  - Github allows developers to share projects with others  **Cons:**  - Xcode is available only on Mac  - Would not be able to access project on any other OS if Xcode is used  - Visual Studio for Mac is expensive | **Pros:**  - Visual Studio is a cross platform IDE that can program Java, HTML, and Javascript  - Bluefish is a powerful HTML editor that also supports other languages  -Bluefish is free and multi-platform  -NetBeans is a free open-source IDE that supports the needed languages  - Github allows developers to share projects with others  **Cons:**  **-** Visual Studio is pricey  - Bluefish user interface is very outdated and not easy  - NetBeans can be unstable and uses a lot of memory | **Pros:**  - Visual Studio is a cross platform IDE that can program Java, HTML, and Javascript  - Github allows developers to share projects with others  -NetBeans is a free open-source IDE that supports needed languages  **Cons:**  **-** Visual Studio is pricey  - NetBeans can be unstable and uses a lot of memory | **Pros:**  - Github allows developers to share projects with others  - Bootstrap is a free open source toolkit that is used to create simple websites that are mobile friendly  -Bootstrap supports HTML and Javascript  **Cons:**  - Bootstrap has a learning curve and may take time  - Bootstrap is slow and has huge files  - Ever mobile device has a different native IDE if used  - If native IDE used, different language experience is needed |

## 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 would be the most suitable to use as the operating platform for “Draw It or Lose It.” It offers compatibility, requirements, and tools that may aid in the application development. Windows is also a very familiar OS and is easier to navigate for many people.
2. **Operating Systems Architectures**: Windows uses a layered system architecture that has two modes, a protected/kernel mode and user mode. There are two versions of the Windows architecture, a 64-bit and 32-bit version. The architecture is made up of different modules. Those modules are a hardware abstraction layer (HAL), microkernel, executive service, environment subsystem, and integral subsystem. The architecture of the kernel mode consists of the HAL, driver, microkernel, and executive services. In kernel mode, the executing code has complete access to the underlying hardware. This mode has the most trusted functions of the OS. In comparison, the programs and subsystems used in user mode are limited to what system resources they have access to. Unlike kernel programs, user mode programs do not have direct access to the hardware. Code that runs in user mode delegates to the system APIs to access hardware and memory. Crashes in kernel mode are catastrophic to the PC. On the other hand, user mode crashes are always recoverable.
3. **Storage Management**: To manage storage on Windows, there are a couple of options. One option is Microsoft Azure. Azure is a cloud-based system that can be used to store and manage data. Another option to manage disk space is by using Windows built in management system, Storage Sense. Storage Sense is mainly used to get rid of unused files.
4. **Memory Management**: A memory management technique that Windows uses that would aid in running “Draw It or Lose It” is Prefetch. Prefetch takes techniques from Paging but reduces the overhead involvement. Prefetch loads important pieces of data from storage into the memory before it is even needed. It does this by using a Cache Manager to monitor the data that is being moved. The Cache Manager builds a map file of the data movement to reference. When the application is opened, the Cache Manager alerts the Memory Manager to use the map file to load the most needed data, like the image library, into memory.
5. **Distributed Systems and Networks**: The “Draw It or Lose It” application will communicate across various platforms by using a client-server system. A client, or platform, communicates with the server for data. Then, the server responds to the client with the requested data to perform. An example of this would be if a player signs in. That action has the client call to the server to verify the sign in information. If the information exists in the server, then the player is signed in. On the other hand, if the information does not exist within the server, the client will show the player that the information does not exist or is incorrect.
6. **Security**: A user security technique that can be used for “Draw It or Lose It” users is to have multiple types of verification. One verification will be a user-made password. The other will be a random generated code. The users will have to register a phone number or email when they create an account. When the user signs in, they will put in their username and user-made password. Once that is verified by the server, a code will be sent to the registered phone or email for verification.