

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)6

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 01/23/21 | Patrick Martin | Initial Draft created. |
| 1.1 | 02/07/21 | Patrick Martin | Evaluation of Platforms |
| 1.2 | 02/21/21 | Patrick Martin | Recommendations |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room’s popular game “Draw it or Lose it” is currently only offered as an Android application. As such, it alienates a large portion of potential customers whom The Gaming Room hopes to recoup. The solution to this problem is the development of a web-based application that will be accessible to users across all operating platforms. The existing logic from the Android app will need to be leveraged and built into a client-server-based model. Webhosting and server usage should be pursued through a third party.

## [Design Constraints](#_2et92p0)

* Infrastructure should be scalable and include the ability to accommodate a growing base of users
* Advertising slots built into the GUI
* Server-side Team (Ruby or Python professionals to build backend support for the application)
* Client-side Team (HTML, CSS, and JavaScript to design the user interface)
* QA professionals needed to test application at multiple stages of development
* Application webhosting plan (Associated licensing fees and costs)
* On call maintenance professional for issues after application is pushed to production
* Draw It or Lose It requirements:
  + Ability to have multiple teams play at a time
  + Each team can consist of multiple players
  + Game and team names must be verified as unique when user inputs selection
  + Unique identifiers for each game, team, and player to ensure that only one instance a game can exist at one time

We are operating under the assumption that because the Android app has been built using Java that most of the code will be reusable in a web application. However, to preserve the requirements, we’ve listed them above in case a rebuild is necessary. Optional advertising slots will be built into the GUI as a means of increasing revenue for The Gaming Room. Most application webhosting plans include sufficient server access and customer service support, but in-house support and additional storage could prove necessary depending on growth. Finally, we intend to use the Agile process and to divide our team up into areas of expertise in order to accomplish a working application as quickly as possible. The teams will include frontend, backend, quality assurance, and administrative groups which will work on separate issues simultaneously.

## [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 contains the main method which instantiates an instance of the GameService which uses the Singleton pattern. This creational design pattern allows for the new object of GameService to be created while ensuring that it is the only such object created. The SingletonTester class was designed to validate that the Singleton pattern is working properly. The GameService object contains getters for each of the three classes (Game, Team, Player) it interacts with. It also contains pointers for those classes to ensure gameplay flows as expected.

The Entity class has been created to allow for object inheritance – which is the mechanism for basing one class upon another using similar implementation. The Game, Team, and Player classes will be extensions of the Entity class. This base class will be used to store common variables and methods for each of the three classes - which improves efficiency in our code by reducing redundancy. The three subclasses can now be built by adding in only the variables and methods that make them unique from one another and the base class.

The cardinality between GameService, Game, Team, and Player is “0..\*”. This indicates that the relationship is optional on one side and could have multiple on the other. For example, a team can be instantiated with zero players, however, a team could also be created with one or many players. A player cannot be created without first having a team. This same relationship exists between the Game and Team classes as well as between the GameService and Game classes.

The use of private variables and getter methods in the Entity and GameService classes is an example of encapsulation. Encapsulation is the act of protecting variables from being directly accessed from outside of the class they belong to. However, GameService is meant to use the Singleton pattern and does not employ any mutator methods. This will prevent users from altering data once an instance has been created.

****

## 

## [Evaluation](#_2o15spng8stw)

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Although it is less common than Linux or Windows, Mac OS X server can be used to host a server. It is available through apple and not incredibly expensive to implement (currently $20). Mac OS X Server will allow The Gaming Room to set up a dynamic website that will permit users to create, retrieve, update, and delete records from its database. | Linux offers many options when it comes to hosting a server for a web-based application. Linux is open source and the options are either low cost or free. The server software will be adaptable to the specific needs of the business however, an experienced Linux administrator will be needed to set up the system. | Windows offers their own server software, Microsoft Azure which leverages cloud computing. The service has a monthly cost of $100 but includes technical support. Windows is the leading non-mobile OS by market share and finding someone with the skillset needed to assist in set up and maintenance would likely prove easier than with Linux or Mac. | Using an Android or iOS device to host server for a web-based application would offer the benefits of mobility. However, the computing power of typical mobile devices cannot keep up with that of larger computer systems. This lack of power may have a negative impact on the end-user experience as most of the computing is meant to take place on the server side for our application. |
| **Client Side** | Mac OS X is not open source. The set up will likely require someone who specializes in OS X which may be expensive and difficult to locate. The amount of time would again depend on the experience level of the individual you hire – but due to a lack of Mac experts, the company would likely be at the mercy of whoever they can find. | While the cost to implement may be low compared to other operating platforms, Linux can be very tricky to navigate. An experienced Linux user would be necessary to develop a distribution system open to multiple types of clients. Again, the pool of candidates with experience in Linux will be shallower than that of Windows or even Mac. | Again, the product offered by Windows is much more expensive, but it includes technical support that will help The Gaming Room mend any issues with clients while not needing to pay for in-house support. Experienced Windows professionals are dime a dozen and the time of set up should be quicker than Linux and Mac OS Server. | The cost of developing an application for mobile is tricky. The user would enjoy “on the go” gaming, which is very popular. There are also plenty of professionals specializing in mobile development. However, Android and iOS are so different from one another that the application would need to be developed twice to reach multiple platforms which could prove costly. |
| **Development Tools** | XCode, Eclipse, and Atom are all popular IDEs that can be used from the Mac OS X operating platform. In order to write an application that works in OS X, you could use Objective-C (which is considered the original native language for Apple) or Swift – Apple’s newer custom language. | Brackets, Sublime, and Atom are all highly rated IDEs that can be used on the Linux platform. The two most prominent languages for writing Linux-compatible code are C and assembly language. | Microsoft Visual Studios would be an obvious choice for developing in Windows, however, there are an abundance of third-party IDEs as well (Net Beans, Eclipse, etc.) The prominent programming languages for Windows include Javascript, Visual Basic, and C. | Popular IDEs for Android mobile devices include AIDE and Droidscript. Most of these Android applications are created using Java. For iOS, XCode offers a decent mobile IDE. iOS apps are written using either Objective-C or Swift (much the same as Mac OS X). |

## 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**:

My recommendation to The Gaming Room would be to invest in machines running the Windows operating system. This would allow them to expand their hit game -   
“Draw It or Lose It” – to users on various platforms by leveraging a distributed system. Windows is the most popular platform by market share and finding expert help would be easier and likely more affordable than with other platforms. Windows also offers a large range of IDEs that will not cost the company too much in initial overhead.

1. **Operating Systems Architectures**:

The Windows platform features a kernel at the core of its operating system which is responsible for carrying out system management tasks. Using this functionality, outside applications can access memory and carry out other processes vital to the application without affecting the processes that allow the operating system to function. In short, The Gaming Room could run test and develop their games without seeing a drop in the machine’s performance.

1. **Storage Management**:

Windows has multiple utilities that come included with the operating platform. Using Disk Management, users can view and manage various aspects of the storage on their machines. Using Disk Clean Up and Storage Sense, Windows can automatically remove unnecessary file from your hard drive. These utilities would make storage management more efficient for The Gaming Room.

1. **Memory Management**:

Similar to storage management, the Windows operating platform also has a built in utility to assist with memory management. The Memory Management utility keeps track of free and allocated memory locations while also determining how and to which processes to allocate memory. This utility would be critical for smoothly accessing the hundreds of images required to play “Draw It or Lose It”.

1. **Distributed Systems and Networks**:

To reach each client on potentially different operating platforms, we must deploy a client-server distributed system. By allowing each client to access the game through a single server location, each client application can be developed to play to the strengths of that operating platform.

1. **Security**:

Windows comes with Windows Defender which should be sufficient for the operating system itself, however, additional security can be added should The Gaming Room choose to. The data traveling between the server and clients would need to be encrypted to protect it from any outside malice. This can be done using SSL which is a transport protocol that establishes secure and encrypted links between machines over a network.