

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 | 9/19/2021 | Jacob Sullivan | -Added Executive Summary and Design Constraints |
| 1.1 | 10/13/2021 | Jacob Sullivan | -Revised Exec. Summary, Design Constraints,  UML Design  -Updated Evaluation Table  -Added Recommendations and Client Requirements |

**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 develop a web-based game based on their current exclusively Android-based game, Draw It or Lose It. The game must have the ability to serve multiple clients on multiple platforms. The game software must also be able to pick and choose from an extensive library of stock images as clues and render them at a steady rate. It must also meet the following specifications:

**Client Requirements**

* Only one instance of a game may exist at once
* No teams may have the same name, names must be unique
* Teams will have multiple players assigned to them
* Games must have the ability to have one or more teams involved

## [Design Constraints](#_2et92p0)

* Game will need increased storage to store new platform’s users, teams, games, and more images that will adequately render on said platforms, as well as new clients to handle communication between these platforms and the server
* Deploying on new platforms will require development, testing and production specific to each platform desired, which may be costly and require hiring or more staff/training of existing staff
* Security on Androids is different to security on other operating systems; new methods of authentication and permissions must be designed to handle each new platforms requests properly while also protecting growing amount of user specific data

## [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 UML diagram ties to this project should tell the client everything they need to know about the application we have designed for them It consists of 7 classes. Entity is our main parent class. This houses the values for ID and Name for use in the classes ties to it. Classes Game, Team, and Player all inherit from the Entity class, making them child classes. This way, they have access to the public “getter” methods in eclipse, giving them access to the values stored in id and name specific to the objects the classes are dealing with.

Encapsulation is used to ensure no classes outside of each class may alter key variables and arrays used, as well as our default constructor in Entity. Encapsulating the default constructor, Entity(), ensures no empty instance of a team, player, or game can be made. Encapsulation of each array ensures only the appropriate class may add to those arrays when necessary.

ProgramDriver makes use of SingletonTester simply to ensure our singleton pattern design was successful in elimination of multiple instances of one game. The client will also see that there is a one to many cardinality between GameService and Game, Game and Team, and Team and Player respectfully.

Polymorphism also plays a role in our child classes, as each makes use of the parent classes functions getName() and getId() in their own manner. Player uses these functions to return information relative to a specific player, while Team uses the same functions to return information about a specific team. Note these are used in yet another show of polymorphism, toString(), which is overridden in each class to provide the proper information requested by the caller.

**"The Gaming Room UML diagram. The top of the diagram is labeled as com dot gamingroom. Test boxes are placed in two layers. The first layer has three text boxes and the second layer has four of them. In the first layer, the 'ProgramDriver' textbox points to 'SingletonTester' textbox. The 'ProgramDriver' textbox contains the text 'asterisk main round brackets.' The 'SingletonTester' textbox contains the text 'asterisk testSingleton round brackets.' The arrow between these two text boxes are labeled 'open two angle brackets uses close two angle brackets'. In the second layer, there are 'GameService', 'Game', 'Team', and 'Player' text boxes. The 'GameService' textbox has texts arranged in two layers. The first layer contains games colon List open angle bracket Game close angle bracket, nextGamesId colon long, nextPlayer Id colon long, nextTeamId colon long, and service colon GameService. The second layer contains GameService round brackets, getinstance round brackets colon GameService, addGame open parenthesis name colon String close parenthesis colon Game, getGame open parenthesis id colon long close open parenthesis colon Game, getGame open open parenthesis name colon String close open parenthesis colon Game, getGameCount round brackets colon int, getNextPlayerID round brackets colon long, and getNextTeamId round brackets colon long. The 'GameService' box is connected with the 'Game' textbox with a line labeled 'zero dot dt dot asterisk'.  The 'Game' textbox also contains text in two layers. The first layers contains the text teams colon List open angle bracket Team close angle bracket. The second layer has Game open round bracket id colon long comma name colon String close parenthesis, addTeam open parenthesis name colon String close parenthesis Team, toString round brackets colon String. The 'Game' textbox is connected with the 'Team' textbox with a line labeled 'zero dot dt dot asterisk'. The 'Team' textbox also contains text in two layers. The first layers contains the text players colon List open angle bracket Player close angle bracket. The second layer has Team open parenthesis id colon long comma name colon String close parenthesis, addPlayer open parenthesis name colon String close parenthesis colon Player, and toString round brackets colon String. The 'Team' textbox is connected with the 'Player' textbox with a line labeled 'zero dot dt dot asterisk'. It contains the text Player open parenthesis id colon long comma name colon String close parenthesis and toString round brackets colon String. The 'Game', the 'Team, and the 'Player' boxes point to the 'Entity' textbox in first layer. The 'Entity' textbox contains text in two layers. The first layer has the text id colon long and name colon String. The second layer has Entity round brackets, Entity open parenthesis id colon long comma name colon String close parenthesis, getId round brackets colon long, getName round brackets colon String, toString round brackets colon String.**

## 

## [Evaluation](#_2o15spng8stw)

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | -Excellent security, not much malware exists targeting MacOS; few vulnerabilities  -streamlined, user friendly GUI  -app would be easily translatable to iOS due to similarity  -high cost for a server capable of running an application of this nature, also must purchase new hardware to stay current  -advised to have dedicated network professional to handle any issues | -cheapest option  -highly customizable, scales well, and very malleable  -well developed permissions system allows for potential high security  -compatible with industry-standard web hosting software  -support for specific versions volatile, ranging from a few months to many years  -steep learning curve for unfamiliar developers  -no dedicated support organization, based on company you purchase hosting from | -streamlined, user friendly GUI  -access to Microsoft support/security applications  -access to APS, .NET, and IIS software framework tools designed for streamlining web-based app development  -high licensing fees for server  -minor chance of running into compatibility issues as opposed to Linux  -average security, most existing malware designed to attack Windows | -Generally not used in hosting web-based application servers  -Mac, Linux, Windows server based websites will run on mobile browsers, but may experience compatibility issues  - app can be developed and uploaded to respective platforms app stores, eliminating need for web-based communication and allowing use as a native app  -fees attached to deploying game on app stores, as well as a lengthy approval process  -growing focus of hackers on mobile platforms and exploits will require large focus on security |
| **Client Side** | -mid-level expertise required, platform already very popular and commonly used, familiarity high  -moderate development time, applications like Xcode available to assist in process of altering code to run properly on macOS  -low to moderate cost dependent on employed devs familiarity with platform and necessity of training | -mid-level expertise required, platform already very popular and commonly used, familiarity high  -low to moderate development time due to common use and many development apps available  -low to moderate cost dependent on employed devs familiarity with platform and necessity of training | -low to mid-level expertise required, platform already very popular and commonly used, familiarity often high, Microsoft support available if not and highly regarded  -low to moderate development time due to common use and many development apps available  -low to moderate cost dependent on employed devs familiarity with platform and necessity of training | -mid to high-level expertise, dependent on choice to solely access via web browser or develop as app to be sold on app stores as well as support for various versions of mobile platforms in circulation  -moderate to high cost, dependent on choice between web-based devs and more niche native app devs  -low to high development time; app already exists on android, mac and ios very similar cutting down on dev time; high only if client chooses to develop as app to be sold on app stores |
| **Development Tools** | Languages:  -Java  -C/C++  -Swift (apple excl.)  -Python  -HTML  IDE’s:  -Xcode (for iOS)  -Visual Studio  -Adobe Dreamweaver (recommended for HTML)  -many more IDE’s available to choose from, including some with licensing fees such as WebStorm, Native React, and IntelliJ, that are used in cross platform development | Languages:  -Java  -C/C++/C#  -Python  -HTML  -JavaScript  IDE’s:  -Visual Studio  -Eclipse  -Atom  -many more IDE’s available to choose from, including some with licensing fees such as WebStorm, Native React, and IntelliJ, that are used in cross platform development | Languages:  -Java  -C/C++/C#  -Python  -HTML  -JavaScript  IDE’s:  -Visual Studio  -Eclipse  -Atom  -many more IDE’s available to choose from, including some with licensing fees such as WebStorm, Native React, and IntelliJ, that are used in cross platform development | Languages:  -JavaScript  -Objective-C  -Python  -Swift (iOS excl.)  IDE’s:  -Visual Studio  -Android Studio  -Xcode (for iOS)  -development of mobile apps is performed on non-mobile platforms and ported to mobile. Additional hardware and software may be required. May also require an additional development team focused on mobile platforms if releasing app as a native app on app stores |

## 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**: I recommend using Linux as the operating platform for server hosting. This operating system is widely used, and although it has a steep learning curve, is touted for its ease of use once familiar. It’s highly attractive in this scenario because it’s the cheapest option of the three, even free in some cases. However, a lot more responsibility will fall onto the development team. Funds will need to be allocated for training, and certain developers allocated or hired for upkeep and support. The support your company will get for Linux will come in some small capacity from the company you purchase the server from and will depend on the price and package your purchased. However, if your developers are competent, hard working and well trained, they should have no issue using the vast online resources to handle any issues with the system that may arise.
2. **Operating Systems Architectures**: Linux possesses an architecture that includes the kernel, system library, hardware layer, system, and shell utility. The kernel, which has been developed for over 30 years via contributions from developers all over the world. This kernel supports the installation of a very wide range of hardware, boasting a superior compatibility to its competitors. It’s multiuser, which allows your development team to use applications, memory, and RAM simultaneously. It also carries a very high security reputation, featuring various methods of authentication and security applications resistant to the majority of malware that Windows is vulnerable to.
3. **Storage Management**: Amazon S3 would be the best option for this application. It’s cost efficient, at $3 for the first 50 terabytes per month and only $0.023 per gigabyte after that. It’s also the leader in its field, featuring on demand scaling, high durability, superior performance and high-level security.
4. **Memory Management**: Linux features a memory management subsystem built into the operating system. Its functions include implementation of virtual memory and demand paging, memory allocation for kernel internal structures as well as user space programs, and file mapping into address spaces to name a few. It features many configurable settings and only requires a small amount of training to become familiar with, as it does feature its own special “jargon”.
5. **Distributed Systems and Networks**: Being that we will utilize Amazon S3 as our cloud-based server, this eliminates the possibility of an outage affecting our data as there will be no dedicated hardware at a location holding all the needed files and information gathered to run the game and properly store profiles of players and teams and so on. In most cases, being an application with cloud-based storage, any connectivity issues experienced by users are likely to be due to connection speeds on the user end. In the even that an event occurs in which many issues are raised at once and a problem is detected, AWS offers a variety of tools and customer support ready to quickly diagnose and repair these issues. Amazon also offers encryption features and access management tools to prevent unauthorized access to your server.
6. **Security**: One of the major benefits of Linux as an operating system is its well-known high level of security as compared to windows. Much of the malware currently in existence has been designed to attack Windows systems, as they have been around and most widely used for years. Linux also features several configurable security settings, and several security focused distributions of Linux are readily available. AWS, as mentioned above, also features encryption services and several security tools, useable at your disposable, to help protect your server. It is advised that authorization of users and a strict permissions system, advisably more complex than “ADMIN” and “USER” as it sits currently in your code, should be implemented as an added measure.