

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 | <9/19/21> | Eric Singleterry | Draw It or Lose It initial draft & mockups |

**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 is creating a game called Draw It or Lose It where the user will have 45 seconds total to correctly guess which random image is being displayed. Initial challenger will have 30 seconds while the drawing renders and the opponents will have an additional 15 seconds once the drawing is fully displayed.

## [Design Constraints](#_2et92p0)

1. Mobile App compatible on IOS and Android
2. App available on Apple App store and Google play.
3. Users and Teams must be established

## [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 Directional Diagram describes how the Game, Team, and Player classes are utilizing the Entity class for its information. The Entity Class is utilizing its “universal” variables to initialize the communication of the game, teams within the game, and the players within the teams.

**"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)

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** | Utilizing Mac OS X Server for Web Hosting can be beneficial since it runs on an Apache sever, making it a quick and reliable server. GUI interface for setting up the hosting sever is also easy to understand for a price of $19.99 | Utilizing Linux will keep costs low and support can also come from open source forums. | Utilizing Windows severs varies in cost depending on the platform. Hosting through HostGostor ranges between $5.00 – $15.00 monthly. Hosting through AWS can also lower cost, since its based off usage and scalability. | <Evaluate Mobile Devices for their characteristics, advantages, and weaknesses for hosting a web-based software application.> |
| **Client Side** | Users are able to download a Mac OS Sever into a personal Mac computer. Creating a local host users can set up and manage multiple Mac computers for the same price of $19.99 | Users are able to local Host Linux OS via an old computer that can keeping cost low. A new dedicated server can also be purchased if the client is expecting rapid growth. | Users are able to local host Windows servers too, but costs will also remain higher. | <Determine the software Udevelopment considerations (cost, time, expertise) that are necessary for supporting multiple types of clients as they pertain to Mobile Devices.> |
| **Development Tools** | Swift will be the language of choice for Mac OS with developer tools such as Homebrew for utilizing the power of Unix in the Mac platform. Developers will also need and IDE such as Xcode or Atom to build and run their code. | Linux being as open sourced as it is, there are a plethora of tools a developer can use. Devs can utilize Visual Studio Code, Sublime, or Eclipse to script Java and / or C++ | Developers are also able to utizlize a multitude of dev tools such as, Visual Studio Code, Sublime, or Eclipse to script Java, C++, HTML | Utilizing servers on AWS developers can compile their services through one means. They can develop for both Android and IOS. |

## 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**: The recommended Operating Platform is Windows OS since the project is a Web based system. Developing on the Windows platform enables the widest range of users to interact with the platform. This will also enable testing to focus on most users and spot testing can be performed for Mac OS.
2. **Operating Systems Architectures**: Utilizing the Windows OS enables users to have a consistent interaction with the Web application by utilizing the Google Search Engine. Creating a consistent user interface and storage such as cookies can create for a more precious caching system to provide quicker response times for the user.
3. **Storage Management**: EBS storage management can be used with SSD and HDD based storage systems. Large storage files can be storage using the slower HDD storage while the game interaction scripts can run on SSD based storage.
4. **Memory Management**: A minimum of 8Gbs of memory can be used to run the Web Application, but the ability to scale up to 38 Gbps will be using when multiple users attempt to run the system.
5. **Distributed Systems and Networks**: Utilizing AWS hybrid Virtual Private Cloud (VPC) with Client-Side Encryption will enable the Web Application to run on low cost with a rapid access to users by using their EC2 servers and CloudFront.
6. **Security**: Keeping the encryption keys on the client side will provide a more secure system. We can also establish both Public and Private VPCs for the we application. User data such as PPI and PII will be stored in the Private VPC while the game files that users interact to load a game will be stored in the Public facing VPC.