

# **Draw It or Lose It**

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template** 1](#_Toc115077317)

[**Table of Contents 2**](#_Toc115077318)

[**Document Revision History 2**](#_Toc115077319)

[**Executive Summary 3**](#_Toc115077320)

[**Requirements 3**](#_Toc115077321)

[**Design Constraints 3**](#_Toc115077322)

[**System Architecture View 3**](#_Toc115077323)

[**Domain Model 3**](#_Toc115077324)

[**Evaluation 4**](#_Toc115077325)

[**Recommendations 5**](#_Toc115077326)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 03/17/2023 | Robert Cook | Working game, team and player classes. Added new class, Entity. |

## [Executive Summary](#_sbfa50wo7nsh)

The gaming room wants to create a web-based game that can run on more than multiple platforms than just android, the game is called “Draw It or Lose It”. This game will have multiple teams that have several people on each team. There will be an image pulled from a set library of images and each team will take turns guessing and each will have 15 seconds to make their guesses.

## Requirements

*<* Please note: While this section is not being assessed, it will support your outline of the design constraints below. *In your summary, identify each of the client’s business and technical requirements in a clear and concise manner.>*

## [Design Constraints](#_2et92p0)

There will need to be one or more teams involved that has multiple people. The Game and Team names must be unique and needs to check if the name is free or in use. Only one instance of the game can be open at a time. It must also run on multiple platforms. It already runs on android so we need to work it into other mobile platforms as well as Windows, Linux, and Apple. In order to get it running on Apple we need to be able to write it in swift or come up with a way to use existing code to be run on other devices.

## [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)

In the diagram below you can see that there are seven classes: programDriver, SingletonTester, Entity, GameService, Game, Team, and Player. Think of the Entity class as a parent interface and then GameService, Game, Team, and Player as four child classes. These four classes relate to each other in an association relationship where each one depends on the other. The singletonTester class inherits from programDriver, which is the main class. Inheritance is the most object-oriented programming principle in this diagram. It has enabled the singletonTester class to perform activities and responsibilities inherited from the main class.

**"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** | This is popular in web hosting. Flexible terminal commands to configure the server, access, or make changes.  The advantages of it are that it is upgradable and has various options for different web hosting requirements. The disadvantage are that it is less preferred for web hosting services. | This is secured and most preferred.  Advantages of it would be that security flaws are caught before they become a problem and it is the most preferred for web hosting services.  A disadvantage is that it is difficult to find application for the web hosting required needs | There is more software available compared to other OS. This is dominant to the other platforms.  Advantages of it are that there are high resource requirements, less loading times, and high comfortability.  Disadvantages are that it is very susceptible to viruses and poor technical support. | This is better if the server is immobile and can be tracked in a single place. This is more popular and has high portability.  Advantages are that is has a wider reach, better compatibility, and is cost effective.  Disadvantages are that it has poor security and is highly selective to various smart mobile devices. |
| **Client Side** | Requires moderate expertise and time required to develop the software.  Client would be charged monthly. Much time would be needed to access the software. Cost is similar to windows. | Requires high expertise since there are few applications available.  Less loading time.  Since it is not as popular it is more expensive. | Has high resource requirements but would require less expertise.  Less loading time.  Since more resources would be required it would be expensive. | Cost-effective.  Takes less time to load a page.  Common so it has high technical support for clients. Provides flexibility to clients and developers to see updates at any place. |
| **Development Tools** | We can use swift to run languages and notepad++. We can use HTML, CSS, and JavaScript languages however we are not limited to these. Frontend languages we can use Java, Python, PHP, and Ruby. | We can use visual studio, eclipse, and notepad++ as well as many others. Languages consist of HTML, CSS, and JavaScript. Java, Python, PHP, and Ruby can be used to create the frontend. | More user friendly but can use the same languages as Linux. | Android Studio or Swift for the IDE. Languages consist of but are not limited to HTML, CSS, and JavaScript. Java, Python, PHP, and Runy can be used for the frontend and general purposes. |

## 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**: Recommend starting with Windows as it is common in developing web-based software, highly secured with less loading time, and it is relatively cheap. It is also compatible and portable. It also integrates with current Android build. You can install Android Studio, or choose a cross-platform like Xamarin, React, or Cordova.
2. **Operating Systems Architectures**: Windows offers services provided by all Windows-based applications that enable them to show a GUI (Graphical User Interface) while accessing system resources and more. These also refer to Graphics and Multimedia, messaging, and web services and can be used using a user account or a server. Windows separates its operating system into User mode and Kernel mode. User mode processes are user facing and affect much of what the user interacts with. Kernel mode is more under the hood and low level, dealing with inputs and outputs, memory management, networking, hardware management, and routines. Windows utilizes a directory structure to hold data. Windows also supports multiprocessing and modularity of hardware, allowing for customization of the system.
3. **Storage Management**: Database management system is the best storage system that works efficiently and effectively with Windows. It is easy to use and runs on multiple operating platforms and is highly adaptive. However, it is recommended that you use Microsoft Azure due to competitive pricing, customer support, and continuous updates. Other features included are Docker containers which can be used to leverage cloud storage instances, Cloud based storage which will allow for scaling up or down depending on user base, and multiple storage options (Azure File system, Azure Storage Containers, and Azure Blob Storage).
4. **Memory Management**: Windows applies memory compression technique which will accommodate the heavy use from the Draw It or Lose It software. This will increase the responsiveness of the operating system.
5. **Distributed Systems and Networks**: Using LAN as the network, the distributed system will use hubs to connect to multiple computers so that when on crashes the game will still operate. The hub will also serve as a repeater to amplify the signals that deteriorate when traveling a long distance. Have LAN and HUB as the network and the connectivity hardware it will help the system have a small outage overall.
6. **Security**: Windows comes with a built-in security protection software. Due to the high security capabilities for Windows operating platform the user protection against intruders will be higher. Since the system scans for malware, viruses, and security threats it all happens in real time and the system updates in order to fight off theses threats. Azure is also an option since it allows for simplicity in the logistics of security of user info and personal data. Each device can connect to the “Azure App Service” which will use Azure Active Directory for logging in. Other features of Azure include: IP configurations, option for storage in a VPN within the Cloud, can make the database IP Whitelist access only, have a password, and require SSL connectivity to protect the users data.