

Draw It Or Lose It

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 | 08/01/2023 | Ryan LeFebvre | Reviewed and filled out the recommendations portion of the design document. |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room has tasked us with helping develop a web-based video game. Currently, the game is only available on Android, and the Gaming Room wishes to branch out to multiple platforms. The game is called “Draw it or Lose it” and the purpose of this game is to solve puzzles by identifying pictures shown by the computer. The game is set up into several teams of multiple people. When a picture is pulled, the team guesses until time runs out. If time runs out, the opposing team then gets to guess for 15 seconds.

## Requirements

There are several requirements that must be met including:

* Must be played with multiple teams
* Each team must have multiple people
* Only one instance of the game can exist
* Must run on multiple platforms
* The names chosen for the game and the teams must be unique
* User should be allowed to search which names are used/available

## [Design Constraints](#_2et92p0)

Following the list above, we must adhere to these requirements when coding the software. This list only really covers the aspects of the game itself, but we still need to consider application development. The Gaming Room wishes for this game to run smoothly on all devices. Currently, the game is available only on Android. This means further development is also needed for other platforms including Windows, Apple, and Linux. To accomplish this, we will either need to rewrite all existing code into a different language to make it compatible with different machines, or we must find a way to have all existing code run on different devices by inheriting from other languages.

## [Domain Model](#_8h2ehzxfam4o)

The new class we created “Entity” creates a relationship with the Game, Player, and Team classes. This means that these classes inherit or get all their information from the Entity class. The diagram below represents this inheritance visually. Each class contains common references including “id” and “name”, this makes Entity a super-class. The UML model also displays what is called aggregation. This means that an instance of one class has a reference to an instance of another class. When viewing the diagram, we see that GameService has a reference to Games, Games has a reference to Teams, and Teams has a reference to Player.

**"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** | Mac has flexible terminal commands to help configure the server as well as make changes.  Popular with web hosting.  Are upgradeable, various options for different web hosting requirements.  Less preferred for web hosting services. | Flexible terminal commands help configure the server as well as make changes. More cost-friendly than Mac.  Secured, most preferred.  Security flaws are typically caught before they become an issue, is the preferred choice for web hosting services.  More difficult to find supportable applications. | Has more software available compared to other Operating Systems.  Much more dominant than other platforms. Closed platform.  High resource requirements, less loading time, more ease of use for the user.  More susceptible to viruses. Poor quality tech support. | Easier if the server is immobile and tracked back to a single place.  More popular, high portability.  Reach more users, better compatibility, more cost effective.  Poor security. Lots of different brands of cell phone from iPhones to Google etc. |
| **Client Side** | Requires moderate knowledge of operating platform from user. Cost is like Windows. | Requires maximum knowledge of operating platform from user. Cost is typically cheaper. | Requires minimal knowledge of operating platform from user. Cost is like Mac. | Provides more flexibility to users and developers alike. |
| **Development Tools** | The most popular language used on Mac devices is Swift and Mac also contains many suitable IDE programs. Macs can also run many other programming languages including Java, Python, HTML/CSS etc. | The most popular IDE’s available for Linux include Visual Studio and Eclipse, and can be used to develop in many languages including Java, Python, HTML/CSS etc. | Easier to use than Linux in almost every way but is the same in terms of available IDE and languages. | Android and Swift support multiple programs and platforms. |

## 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 that the Gaming Room go with developing a Windows platform. Windows has more software available for use, has the highest ease of use, and there are multiple IDE’s which are compatible with Windows machines.
2. **Operating Systems Architectures**: Windows can enable applications to show Graphical User Interfaces (GUI) and allows the user to access system requirements and resources. These services can be used on a user account or server specifically.
3. **Storage Management**: Windows comes with a feature called storage sense, which allows the user to manage files on their hard drive while also determining how much space they are taking up. Another great feature allows the user to choose where files are saved onto the computer, allowing the user to access these files much easier. Windows also provides cloud storage, meaning files will be backed up off site in the event of catastrophic emergencies.
4. **Memory Management**: Completing this game will require a database of likely thousands of pictures. Memory allocation through windows allows for photos to be saved outside of the default picture folder. This allows the project to be secured virtually anywhere on the machine.
5. **Distributed System and Networks:** I suggest starting with development for Windows-based platforms initially, but it's important to keep in mind that we will eventually need to cater to all platforms, regardless of their operating system or devices. To achieve this, a cross-platform game development tool would be the perfect solution for our project. After some research, I discovered that 'Unity' is a highly recommended and cost-effective game engine that supports a wide range of operating systems, including Windows, Linux, Android, and even i0S. In addition to selecting the right game engine, it's crucial to carefully choose a reliable server. Since multiple users may be playing the game simultaneously, we need a network and server that can handle the high volume consistently.
6. **Security**: Windows comes with built-in security software. It would be recommended to secure projects on another source as well, however. Still, the system comes equipped to scan for malware, viruses, security threats etc.