

Draw It or Lose It

# **CS 230 Project Software Design**

Version 3.0

## 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 |
| --- | --- | --- | --- |
| 3.0 | 10/15/22 | Anthony D’Angelo |  |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room company would like to develop a web-based game that can run on multiple platforms. The game will be called “Draw It or Lose It” and is currently only available on android. The purpose of this game is as follows:

* Multiple teams consisting of several people go four rounds at a minute each.
* When a picture is pulled from a library of images one team guesses till time runs out.
* If not answered each opposing team member gets to answer till 15 seconds runs out.

## [Design Constraints](#_2et92p0)

* One or more teams needed
* Each team has multiple people
* Game and Team names must be unique to allow users to check whether the name is in use or available
* Only one instance of the game can exist at any time
* Needs to run on multiple platforms

These are the requirements needed to follow while writing the code and software. The Gaming Room would like this to run on all devices (Android and iOS). We already have it on android, but we need to work it into another mobile device along with machines like Windows, Linux, and Apple. To do this we will need to find a way to either re-write the code in swift for (Apple devices) or come up with a way to use existing code to be run on other devices by inheriting other languages.

## [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 Entity class creates a relationship between Game, Team, and Player class. This means they all inherit or retrieve information from Entity. With the UML diagram we can show this with inheritance. Each class will share common references like “name” and “id”. This makes Entity a superclass. When we look at their relationship, we see Team and Player is a “has a” type. While Game has a Team and GameService has Games. When we use UML, we call it aggregation (HAS-A). When a user “has a” this means it is an instance of one class and has a reference to an instance to another class. When we look at this diagram, we see GameService has a reference of Games, Games a reference of Tea, and Team a reference of 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** | Easy terminal commands  Popular web hosting  Advantage: Easily upgradable  Disadvantage: Less preferred web hosting services | Easy terminal  Cost efficient  Advantages: Secure, most preferred web hosting services  Disadvantages: Difficult to find apps to support web hosting | More software available  Dominant to other platforms  Advantages: Less loading time, high comfortability  Disadvantages: Easy to get viruses | Better if the server is not mobile.  More popular  Advantages: Better compatibility, cost efficient  Disadvantages: Poor security |
| **Client Side** | Moderate expertise and time required. Cost similar to Windows | Maximum expertise and time required. Minimum cost. | Minimum expertise and time required. Cost similar to mac. | Provides flexibility to clients or even developers to see updates at any place. Slightly more difficult to implement than other devices. |
| **Development Tools** | When running languages on macs we can run swift, the more popular option, while mixing in nice tools like notepad++. Though Macs can run all languages. Languages: HTML/CSS/JavaScript while supporting libraries to support the frontend and general-purpose languages. These can be Java, Python, PHP, and Ruby. | Linux can work with visual studio, eclipse, along with notepad++. Languages: HTML/CSS/JavaScript while supporting libraries to support the frontend and general-purpose languages. These can be Java, Python, PHP, and Ruby. | Easier to use than Linux but can run the same. So visual studio, eclipse to name a few. And with multiple tools notepad++ is a simple to use the tool. Languages: HTML/CSS/JavaScript while supporting libraries to support the frontend and general-purpose languages. These can be Java, Python, PHP, and Ruby. | You can create countless apps using android and swift. Both languages and software can be run on all three machines. Languages: HTML/CSS/JavaScript while supporting libraries to support the frontend and general-purpose languages. These can be Java, Python, PHP, and Ruby. |

## 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 would recommend The Gaming Room starts on windows devices as it has more software available along with minimum expertise and cost to get projects going.
2. **Operating Systems Architectures**: Windows provides services used by all Windows-based applications that enable applications to show a Graphical User Interface (GUI) while accessing system resources and much more. These applications also refer to Graphics and Multimedia, messaging, and web services. These services can be used using a user account or a server specifically.
3. **Storage Management**: Windows 10 comes with a nice feature called storage sense. This allows you to manage files on your hard drive, along with how much space it takes up. Other features include being able to choose to save locations for apps making them easier to find. And just like other drives, you can also use the cloud to save data. The built-in storage system allows for easy file creation and placement for large projects, so they won’t get lost or carelessly deleted.
4. **Memory Management**: You will need to create a database or library with lots of pictures. The memory allocation allows for easy storage of pictures outside of the default picture folder. This allows you to keep your whole project together in a more secure area on your computer. This includes when you’re working with your IDE and opening files from it to create the game.
5. **Distributed Systems and Networks**: I found Develop 4 which enables cross-platform game creation. It’s an IDE that can be run on any device. Once the game is created you can simply export the game file into the web, iOS, Android, and many more options that will allow cross-play. This will help with dependencies. To prevent other problems like outages or connectivity, the company will need to make sure their servers are strong enough to support large player volumes along with backup power for power outages.
6. **Security**: Windows comes with built-in security protection software, but to secure user data and information it would be recommended to use another source. If we are talking about what is on the machine, Windows comes pre-equipped with protection. This system scans for malware (malicious software), viruses, and security threats. This all happens in real-time, and because threats change the system updates automatically to keep the system and user information safe