

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 | 07/15/2023 | Rutger Vaskis | Initial draft |
| 2.0 | 07/30/2023 | Rutger Vaskis | Evaluation |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room is wanting to develop a new web-based game that will be able to be run on multiple platforms for its users. This new game will be called “Draw It or Lose It” and it is currently only available to Android users. The main purpose of this game is to have multiple teams of serval users playing four rounds and each round is a minute long. Whenever a picture is pulled from the library of images, one team will have to guess what the image is until the one-minute timer runs out. If that team does not get the correct answer within that minute, then the opposing team will have a chance to guess the image for 15 seconds. If the opposing team answers correctly, then they will get a point and end that current round.

## 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)

In order to begin to make “Draw It or Lose It” more accessible to users, we first need to layout a design plan. We will need to determine if one or two teams are going to be involved in the coding for the different platforms. We will also have to keep in mind that for each platform, there needs to be multiple users on each team. Each Game and Team names need to be unique from one another to allow the users to check whether a name is available or already in use. Users can only play in one instance of the game at a time, and we also need to configure all of this to be used from multiple platforms.

## [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)

Looking over the UML Diagram, the Entity creates a relationship between the sub-class Game, Team, and Player. Each sub-class gets its information from the Entity class. We are able to show how this is gathered by using the UML Diagram with inheritance. Each class shares a common reference such as “name” and “id”, making the Entity class a superclass. Looking at each of their relationships, we can see that Team and Player is an “aggregation” type. As for Game has a Team and GameService has Games. As we look at the diagram, Gameservice has a reference of Games and Games is a reference of Team, as for Team is 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)

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** | Has Flexible terminal commands to configure the server or to make changes.  Very popular characteristics for web hosting.  Can be upgradeable and has a wide variety of options for web hosting.  Less preferred for web hosting services. | Same flexible terminal commands and configuration as Mac but more cost friendly.  Most preferred for secured characteristics.  Security flaws are noticed before they become a problem, Most preferred choice for web hosting services.  Hard to find applications that support web hosting requirements. | Large amount of software available than other OS.  Dominant characteristics compared to the other OS, close platform.  Has a high resource requirement, less loading time, and is easy to understand and use.  Virus susceptibility is easy. | Having a mobile server is not as good as having a stationary server, the specifications in the other OS are better to use.  Popular characteristics with a high portability.  Gives a wider reach to more users, better compatibility, and cost-effectiveness.  Poor security for devices and highly selective to smart devices. |
| **Client Side** | Will need a moderate level of expertise and time required. The cost is about the same as if you would use windows base. | Will need a highly level of expertise and time required since Linux is not well used. The cost would be a minimum though. | Will need an average level of expertise and time required. Cost is about the same as the MAC cost. | Will need an average level of expertise and time required. Client side for mobile devices allow more flexibility to see updates, slightly more difficult than other OS. |
| **Development Tools** | Using languages on MAC, SWIFT is the more popular option along with Notepad++. MAC’s can run all languages but mostly consist of HTML/CSS/JavaScript while supporting frontend and general-purpose libraires. These can be Java, PHP, Python, and Ruby. | Using visual studio, eclipse along with notepad++ on Linux makes it an easy-to-use tool when developing. There are many more to use as well. Can also use languages like HTML / CSS / JavaScript while supporting frontend and general-purpose libraires such as Java, PHP, Python, and Ruby. | Very easy to use for development. Can run all the same languages as Linux like eclipse and visual studio along with notepad++ as a tool. Can also use languages like HTML / CSS / JavaScript while supporting frontend and general-purpose libraires such as Java, PHP, Python, and Ruby. | Can use Swift and android to create unlimited amounts of applications. Languages and software can be run on all 3 OS. Can also use languages like HTML / CSS / JavaScript while supporting frontend and general-purpose libraires such as Java, PHP, Python, 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 that The Gaming Room starts on a windows device as it has more software available along with the minimum amount of expertise needed and the cost of the project to get going is low. Another thing is that there is not a shortage of IDEs to work with either.
2. **Operating Systems Architectures**: Windows offers services that are used by all window-based applications that enable the application to have GUI (graphical user interface) while accessing system resources and other things. Some of these applications are referred to as Graphics, web services, multimedia, and messaging. All of these applications can be used either from the server side of things or from a user account.
3. **Storage Management**: Windows operating systems also come with great features for storage management. It allows easy access to manage and scrutinize files that are on your hard drive, along with how much space is available or used. Another great thing about windows is that you are able to select where you want to save certain files and projects. It is very easy to be able to move files around so you can organize large projects and their files in a way that makes sense to you and your team.
4. **Memory Management**: When creating this game, we need to always be thinking about memory management. We will need to create a database or a library that will store a lot of pictures. The memory allocation on windows allows for easy storage of pictures away from the default picture folder. This will allows you to keep the library in the place of your chosen in order for it to be easily accessed and to be more secure when working with your IDE and opening files from the library for the game.
5. **Distributed Systems and Networks**: Since there are a number of different platforms and networks, I went ahead and investigated many ways to publish the game to be able to run on all devices. Develop 4 is a great way to enable cross-platform game creation. It is an IDE that can be run on just about any device and once the game is created you can simply export the game files onto the web, iOS, Android, and any other more options that will easily allow you to enable cross-platform play. This will help with dependency. One thing to remember though, when you are thinking about cross-play, you will need to ensure that your servers will be able to handle such a large number of players utilizing the servers. If not planned out correctly this could cause power outages or even an issue with the player having connection issues.
6. **Security**: Windows comes with a few built-in security protection software that work great but in order to better protect game files and user information, I would suggest that you invest in another source like a physically secure server that is protected from the network and any physical contact. With windows though you are able to run security scans at any time to scan for spyware, viruses, malware, and any other security vulnerabilities that may threaten your game. All of this can happen in real time to protect the game files and user information.