

The Gaming Room

# **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 | 09/18/23 | Larry Shayne McDonald | Revisions made on Cover page, Revision history, Executive Summary. Design Constraints were added, and Requirements were updated. The Domain Model, Evaluations, and Recommendation. |
| 1.0 | 10/1/23 | Larry Shayne McDonald | Revisions made on Evaluations, to show advantages and disadvantages of systems. |
| 1.0 | 10/15/23 | Larry Shayne McDonald | Revisions made Recommendations fields of Operating Platforms and Architectures, Storage and Memory Management, Distributed Systems and Networks and Security . |

**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 project is to develop a web-based game based on their current game “Draw it or Lose it”. However, since this is only available on android, the need is for the game to be served on multiple platforms. The purpose of the game is to have unique names for teams and users. The game will last four rounds at one minute for each round. When the system pulls a picture from the library of images, one team will guess until time runs out. If not answered correctly, each opposing team will get a chance to answer until their 15 seconds run out.

## Requirements

The game will have the ability for one or more teams to be involved, along with each team having multiple players assigned to it. The team names need to be unique and allow users to check if that name is in use when choosing the team’s name. Due to memory constraints, only one instance of the game can exist at any given time. This will be aided by the user creating unique identifiers for each instance of the game, team, or player.

## [Design Constraints](#_2et92p0)

* Could run on multiple systems. This will allow more users to play the game.
* Each team should have multiple players, as this will allow for more users for the game to compete against each other.
* There can only be one instance that exists for the game at any time. This will allow for more users to be involved in the same game.
* The team’s name and usernames need to be unique and must have a way for the users to check if that name is available to use. This will stop the system from having issues when the game is in process and help keep track of the users in the game.

## [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 UML diagram shows how the Entity creates a relationship between Game, Team, and Player class. This allows those classes to get the information from the Entity which shows this as a superclass. We can see how the relationships interact with each other. We can see there is a “has a” type between the Team and Player class. However there for the Game class there is a “has a” type for Team and GameService between them. Per the UML diagram, it is shown as aggregation (HAS-A). When a user “has a” I mean there is an instance of one class and has a reference to another instance of another class that is tied to it. Also with the UML diagram, GameService references Games, Games a reference of Teams, and Team a reference of Players.

**"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** | Server is easily accessible; the graphic interface is easy to use and does have a cost associated with that program. However, the Mac system is not as popular as Linux or Windows to perform tasks. | The Linux system has more server options that are a lower cost. However this would require the user to already have Linux knowledge to be able to use the program. | Since the system of “Windows” are being used by more and more people. This would be a choice that could reach more people that have the knowledge already. | Since mobile devices do not have the same capacity as a computer, having its own server might not be the best option. Having the servers run on the mobile devices are cost effective to get them started. |
| **Client Side** | These costs would be like a windows account but since it is not going to be open source. The time requirements will vary between users due to the knowledge already versus the ones that don’t. This could cause those who don’t have that knowledge will cause them to use more time. | Cost would be little to nothing compared to others. However, since Linux is not commonly used, the user will have to have that knowledge to use the program. They will need to have more than just a basic knowledge as it is difficult for those users already sometimes. | These costs would be like a windows account but since it is not going to be open source. The time requirements will vary between users who have the knowledge already versus the ones that don’t. This could cause those who don’t have that knowledge will cause them to use more time. | The cost would be low and also the experience needed to use the system could be easier as the mobile device is smaller than a computer. Flexible for clients and developers to see updates. It is a little more difficult to implement other devices. |
| **Development Tools** | For writing language SWIFT is commonly used. There are multiple IDE’s that can be used like ATOM. | Eclipse can be used for Linux, and Java is most used with Eclipse. But other languages like C++ can also be used with Eclipse. | For the Windows system we can use Eclipse, Pycharm, or Visual Studio. Here we can use either Java or C++ and Python to create the program. | For writing language SWIFT is commonly used. There are multiple IDE’s that can be used like ATOM. It is like Mac but it has a different user license. |

## 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 operating platform of the Windows environment will allow The Gaming room to expand the game as needed. By using the cloud-based system this would allow the game to run with less shortages as there are no shortage of IDEs to work with.
2. **Operating Systems Architectures**: Microsoft Windows uses a graphical operating system published by Microsoft. This program stores files, run software ad works great to play games and videos and can connect to the internet. This would also allow the game to be used by multiple types of devices if it can download the app.
3. **Storage Management**: Windows 10 can manage files on your hard drive and how much space that is taken up. By using the cloud based options this will also help keep the storage more on the cloud versus using a hard drive and other physical storage devices. For a long-term length of storage, this would be the storage on a hard drive or some physical device for storage. For the game's application memory would be short-term due to the user's session. However, storage of the passwords, usernames, and team names would need to be more long-term as they can login and continue playing later when they are online and connected to the game.
4. **Memory Management**: Windows 10 will also allow storage management of Draw It or Lose it photos and players of the game. It allows all this information to be stored securely in the memory. To make sure the software application memory is managed the best way possible there are a few options we can do. For the program we must make sure we have the memory capabilities to store the 200 photos that are being added to the application. We must also make sure that the memory can handle the different usernames and team names that are being used for the game. This must keep the username and team name unique so none of the program runs into bugs created per the names. For the player we must make sure the program has storage and memory to be able to comment on the player's input. One solution that would be cost effective and save physical space that would still meet the requirements would be a virtual memory. This allows for larger programs to be stored and takes up less space on the servers. This also allows for the physical memory to be saved on a physical disc that routes back to its own physical address. (Operating System - Virtual Memory - Tutorialspoint).
5. **Distributed Systems and Networks**: With multiuser system networks, they typically have a database that is shared among the players where they can interact over the network. Currently this is created from scratch by game developers. The features to include in the game would be to have the game be interactive. This will help the user to be more involved in the game. If it is not interactive, it could lose attraction with users. Using experimentation will help provide the best product for the client. To gain more users for the game it should be challenging, not too hard where no one enjoys the game but at least user friendly to help gain popularity. We would use the processes in place already and can update the program to be downloaded on the Play Station Store and have the PS users play against others online at the same time on larger scale.
6. **Security**: The program would be secure with the built-in security that the Windows systems offer. By having the user names and team names being unique, this will allow some security for the game. For added security, we can have the game require a user-id and password to sign into the game. This would allow the player to not worry about someone playing the game using their name and information. Another way to help with security is to not make it where players list very personal information to be able to play the game. It will also depend on what standards the game will have for security and compliance for the program. Although a different source could be recommended to secure user data and information.

References:

Operating System - Virtual Memory - Tutorialspoint. (n.d.). Tutorialspoint. Retrieved

February 14, 2021, from

<https://www.tutorialspoint.com/operating_system/os_virtual_memory.htm>