

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 2**](#_Toc115077320)

[**Requirements** Error! Bookmark not defined.](#_Toc115077321)

[**Design Constraints 2**](#_Toc115077322)

[**System Architecture View 2**](#_Toc115077323)

[**Domain Model 2**](#_Toc115077324)

[**Evaluation 3**](#_Toc115077325)

[**Recommendations 6**](#_Toc115077326)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 09/22/2024 | Marianna White | The changes were made to the following, the cover page, executive summary, the document revision history, domain model, design constraints, system architecture view and recommendation. |
| 2.0 | 10/16/2024 | Marianna White | Updated the findings under evaluation. |
| 3.0 | 10/18/2024 | Marianna White | Architecture Recommendation. |

**[Executive Summary](#_sbfa50wo7nsh)**

The Gaming room project is to develop a web-based game that serves multiple platform systems based on the current game draw it or lose it, which is currently only available on android. The motive of the game is to have a variety of teams consisting of multiple people going a minute for four rounds each. When a photo is pulled from a library of images one team guesses until the time runs out. If it is not answered from each opposing team member, they will get to answer until 15 seconds runs out.

## [Design Constraints](#_2et92p0)

* Each team needs to have multiple players.
* It must be able to run on a variety of platforms.
* Only one occurrence of the game can exist at any time.
* Both game and team names must be different to allow users to check if a name is in use when selecting a team name.

## [System Architecture View](#_ilbxbyevv6b6)

## [Domain Model](#_8h2ehzxfam4o)

Entity creates a relationship between Game, Team, and Player class. This defines as they are all inherit or receive information from Entity. When it comes to UML, we can display this with inheritance, therefore making Entity a superclass. When looking at their relationship, Team and Player is a has a certain 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 specific type, in an instance of one class and has a reference to an instance to another class. When looking at this diagram, we view GameService has a reference of 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** | Mac does have Mac OS X server available for users. As per Apple’s website, it states that Mc OS X server is only twenty bucks. With this it would not be inexpensive to implement. Mac, is not as popular compared to Linux or Windows for performance of certain tasks. | For Linux, is a bit more interesting compared to Mac, as is has many distributions that has sever capabilities. Linux server would be low-cost and open source, which in-turn provides a lot of resources. Many users are not savvy with Linux, so it is best that users that are more familiar with this software uses it. | For Windows, it offers Windows server. Microsoft’s website maybe costly to implement, however it is more functional. Windows is probably the most used operating system, so it would be a lot easier for users to use Windows. | For mobile devices, since it does not have the power as computers does. Hosting a full-fledged server on one may be the best option to choose compared to the others. However, using mobile devices could be more advantageous in relation to cost or little to none. |
| **Client Side** | The cost would be more-so a Windows setup, as these operating systems are not an open source. The time would depend on expertise depending on the user. Users that are more familiar with Mac would require less time versus those who are not as familiar who would need more time. | Lots of expertise and time will be needed. Linux data is required to use the operating system since the maximum cost as it pertains to Linux users. For the cost to be low since it is an open-source. Linux is not as commonly used and someone/user would be required to allow them to work through, as Linux can be difficult to learn. | Windows would have a Windows setup as these operating systems aren’t an open-source. The time would depend on one’s expertise, as someone/user who is not as familiar would require more time to learn Windows. | For mobile devices, the cost would not be as much, little to none since a lot is not required. The experience should not be as difficult compared to other operating systems since it is easier to learn and use. |
| **Development Tools** | Swift would be a more common language to use to write applications for Mac. With this, there are multiple IDEs that can be used for Swift, such as Atom. | The common languages for Linux would be Eclipse and Atom. Eclipse is primarily used for Java, even though it can support other languages, such as C+. | The languages that Windows uses are Eclipse and Visual Studio, which are both popular IDEs for Windows. They can be to develop in HTML, C+ and JavaScript amongst others. | For Mobile devices, iPhones for instance use iOS application, which is typically written in Swift, although iOS and macOS are different in terms of their different capabilities. |

## 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:** Based on my research, Windows is the best operating platform. It is the most widely used and familiar to most people. There are many IDEs available for Windows, and the overall cost is typically lower.
2. **Operating Systems Architectures:** Windows architecture allows applications to use the platform's kernel processes without directly affecting them. This means applications can utilize Windows' capabilities for GUI setup, memory access, and other vital processes without disrupting the platform's functionality.
3. **Storage Management:** Windows includes Disk Management and Storage Sense. Disk Management is used for advanced storage tasks, while Disk Cleanup and Storage Sense help maintain system storage by deleting unnecessary files.
4. **Memory Management:** Windows has built-in Memory Management as a system utility. We would need to create a database for all the game's image files to ensure easy access by the application.
5. **Distributed Systems and Networks:** A client-server distribution system will be used, where each client application relies on a single server application. This allows each client to be developed to its strengths. A robust server network is essential for multiple clients to connect and play the game together.
6. **Security:** Windows Defender is built into Windows for security. We need to encrypt all data being transmitted to ensure security.

Sources used:

<https://www.wgu.edu/blog/5-most-popular-operating-systems1910.html#:~:text=Microsoft%20Windows.,with%20or%20tweaked%20by%20users.>

[Microsoft Supporthttps://support.microsoft.com › ma...Manage drive space with Storage Sense](https://support.microsoft.com/en-us/windows/manage-drive-space-with-storage-sense-654f6ada-7bfc-45e5-966b-e24aded96ad5)

<https://techcommunity.microsoft.com/t5/microsoft-defender-for-cloud/security-control-encrypt-data-in-transit/ba-p/2201008#:~:text=TLS%20should%20be%20updated%20to,the%20latest%20Minimum%20TLS%20Version.>