

<Name-of-Software-Application>

# **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 | 03/24/2024 | Brandon Dolan | There were changes made to the executive summary, design constraints, domain model, evaluation, and recommendation sections. |
| 2.0 | 4/6/2024 | Brandon Dolan | Made changes to the server side, client side, and development tools table.. |
| 3.0 | 4/14/2024 | Brandon Dolan | Made changes to the recommendations sections. |

**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 based on the current game Draw It or Lose It which is only available on Android. The rules of the game require multiple teams of people going four rounds at a minute each and when a picture appears is to guess the image until one team gets it right. After 15 seconds the next time in rotation will guess.

## 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)

* Must run on multiple platforms
* Be able to support multiplayer and consist of multiple teams
* Only one game per session can exist
* Game lobby and team names will be unique as a way to identify players

## [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)

Entity can create a link between Game, Team, and Player. All 3 sub classes are then able to gather information from the Entity class itself. This would then turn the UML into a superclass through inheritance. Team and Player is a “has a type”, Game has a Team, and GameService has games. In UML terms the aggregation “HAS A” refers to the instances of classes in this sense. GameService, Games, Teams, and Player are connected through reference points in the diagram.

**"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 is fairly straight forward and easy to use. Simple user interface for transitioning users. | Difficult platform but cost friendly. More for experience programmers. | Expensive servers to maintain. However it has a user friendly interface. | The operating system performance depends on the specs of the hardware. |
| **Client Side** | Expensive hardware. Takes some time to learn for transitioning users. Users need more advanced knowledge of the OS. | OS is more time consuming for basic users. Linux is not cost efficient. | More expensive than Linux but much easier to learn and less time consuming. Great for beginners and minimal setup required. | Requires more work to implement but there is more flexibility for the user. |
| **Development Tools** | Languages consist of HTML, CSS, and JavaScript. VisualStuidoCode, MacDown, Sublime Text, and Homebrew. | Linux development may take the form of C/C++, Java, or Python.  Python IDEs are often free, e.g., NotePad++. PyCharm is another popular Python IDE. | Microsoft’s Visual Studio is an immensely popular IDE and offers many plugins and integration options, e.g., Jenkins, TestComplete, etc. | Android SDK is Java based and the most widely used Android IDE is Android Studio which is developed by Google as the official development tool. Android Studio is free to download. |

## 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**: While the application is already available on Android, I would suggest making it available on Windows. The client could have their own download installer or add it to the Steam platform as well. If they have the resources available to manage it having it available on iOS would also be beneficial.
2. **Operating Systems Architectures**: Windows is a graphical user interface designed by Microsoft. It has many features such as internet connectivity, file storage, running software, and playing games and videos. The client could use a backend server as the suggested architecture. Using a front-end server would allow for client-side rendering.
3. **Storage Management**: Windows 10 offers Storage Sense which frees up your storage based on item usability. It would also be preferable to have an SSD for internal storage and on the client side of things they could implement cloud storage.
4. **Memory Management**: In Windows 11 there is a feature called DirectStorage that allows you to load your games at a faster rate. This does involve having an SSD and GPU that can support it. The server on the client side will need a minimum amount of RAM to render appropriately. Windows will allow the user to specify how much memory can be allocated to the program itself.
5. **Distributed Systems and Networks**: Online multiplayer games require a network of sorts for multiuser interactions. This requires a database that is shared among the players, and they interact over this network. Developers can then update this database at their leisure to prolong a game. Cloud technology would also allow the servers to run with company databases and have the game run on whatever platform is integrated.
6. **Security**: While security is built into the Windows environment it is recommended to use another source for security such as an antivirus or VPN software. It will always be an ongoing issue no matter how protected you are. If there is no paywall for the game, you could just require as little signup information as possible when opening an account but data breaches are always a possibility. A firewall would also be highly recommended.