

<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 | 05/20/2025 | Nicolas Morrison | Initial Write- up |
| 1.1 | 06/07/2025 | Nicolas Morrison | Updates to The Gaming Rooms requests. They are interested in releasing their game on multiple platforms. |
| 1.2 | 06/16/2025 | Nicolas Morrison | Updated information: Explain architecture needs for other platforms. Show how to implement the architectures based on the client’s needs. Also corrections made to Server-side applications. |

**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 client wishes to create a web-based game that can be played on multiple platforms based on their game Draw It or Lose it. The client’s current game is only on the Android app store. Our team is charged with developing the environment for the web-based game. The client must know that we will only be using JavaScript to accomplish this task. The client also expressed certain requirements such as multiple teams playing and each team will have multiple players. The requirements are listed below.

## Requirements

1. Each game session will have the ability to have one or more teams play.
2. Each team will have multiple players.
3. Each game and team will have unique names to ensure users know whether that team name is in use.
4. Only one instance of the game can exist in the memory at any time.

## [Design Constraints](#_2et92p0)

Since the game is being developed on a web-based distributed environment. It will be difficult for a multiplatform release like the client wanted. Considering our team has limited experience in multiple platforms, we will have to break the team up into smaller teams to develop it for multiplatform. This could potentially cause delays.

## [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)

As you can see from the model, the Entity Class is the super class of the Game, Team, and Players classes. Game, Team, and Players classes are the child classes, and all inherit the methods and attributes of the Entity class. While all the child classes inherit the methods and attributes of the super class, each child class also has unique methods and attributes assigned to them as well. The GameService class is used to ensure the clients’ requirements will be met by providing a one game instance at a time, unique team names (id), unique game name (id), and unique player name (id). The ProgramDriver class will host the main statement and uses the SingletonTester class. The Game class will hold the team list; the Team class will hold the Player List. The Player class does not contain a list; however it will ensure each player has a unique id that can be assigned to a team. The Player class can be on a team, and the Team Class has players, the Player class does not have a team or a game.

**"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** | MacOS does not have a server license available. It’s against their policy to be used as a hosting platform. Therefore, this is a not a viable option. | Linux is an extremely customizable OS. The pros for using Linux is that developing a web-based application can be done with ease due to it’s many languages it supports. The biggest drawback is the minimal choice in pre-built machines and it’s file formatting issues. | Windows has a big advantage when it comes to the amount of users that use it. Most companies, everyday users, and developers tend to use Windows machines . Windows authentication and Active Directory make corporate use easy to set up because it’s ready to go out of box. The biggest con to Windows is simply it's security protocols are not that great compared to Linux and MacOS. | Mobile devices cannot be used as a Server side host. Mobile are not a viable option in this regard. |
| **Client Side** | A pro would be that once the team has experience with Mac OS, it will be easier to use for the client. The biggest con to an Apple based app is that it’s non-accessible unless you have an Apple product. That con alone could really limit the developers and possibly not meet the clients requirements. | Linux provides a cheap and more customizable option for the client due to it being opened source OS. However, since it’s open source. There could be some potential security issues. | Window’s is cost efficient and has tons of options when it involves pricing. Windows also has great technical service support. The downside is that you’ll need a good System Administrator who specializes in Windows to keep things running smooth, which could be costly. | The pros for mobile is that many applications and tools are available at varied price points. This allows for more shopping. With that said, the tools that are available usually lack the full features of their PC counter parts and many mobile devices have specific OS’s that they cater to. This would make accessibility an issue. |
| **Development Tools** | Mac uses the Swift programming language. The tool available for MacOS and IOS developers is Xcode and Xcode Cloud. Xcode cloud is a service that was designed specifically for Apple developers. | Linux offers a plethora of languages and development tools to choose from. You have Visual Studios, Eclipse, NetBeans, Kdevelop, and Docker. | Windows is primarily written in C. With that said, Visual Studios is the primary development tool and it’s an amazing one at that. It sort of is an all-in-one package because it used for an IDE, code editor, and source and version control. Visual Studios is an amazing development tool and easy to use. | Mobile phones typically use Java as the language. Since Java is an object-oriented language, it makes it a great choice for many developers. Some phones also allow you to use Python and C++ or other C variants. C variants are typically used in the gaming sphere. |

## 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**: Our team finds that Window’s is the recommended operating platform for the web-based game. With its ease of access, affordability, and ease of development tools. Windows would be the most optimal choice for multi-platform game development. Considering most of the world (around 90%) uses Windows, Windows is the optimal choice for the operating platform.
   1. Windows easily integrates with current Android build
   2. Offers an extensive amount of Microsoft products to make the design, development and deployment easier
   3. Has a excellent reviews with a large userbase
2. **Operating Systems Architectures**: Our team recommends using a hybrid architecture. Doing so will ensure the client has the best features from monolithic, layered, and micro-kernel architectures. Specifically, our team recommends using Microsoft Windows NT kernel. Choosing a hybrid architecture over a singular architecture is optimal because its layered approach is easier to manage. The hybrid architecture provides better security and protection than a  monolithic, layered, or micro-kernel. Another reason to choose a hybrid architecture is its ability to be customized based on the client's needs.
3. **Storage Management**: Microsoft Azure is what our team recommends for the project. We recommends this because Microsoft works with Windows seamlessly, its affordability, expert and available customer support, and consistent updates. It’s the optimal choice. Other reasons include:
   1. Azure Cloud allows use of Docker containers which leverage cloud storage instances that can be deployed to Azure Cloud.
   2. Scalability options for storage
   3. Cost efficient
   4. Since the client requires 1.6GB due to their 200 eight-megabyte images. Azure officers file system, storage containers and blob storage which gives the client added access to additional storage options.

**Citations:** **https://azure.microsoft.com/en-us/services/storage/files/#features**

1. **Memory Management**: Microsoft has a storage and memory management system that is called Azure Storage. The OS also includes virtual and physical address space for memory allocation. The client can also consider using OneDrive, Visual Studio or Azure Cloud services to store or manage versions. Windows also will continue to improve upon their newer releases to improve memory management capabilities.
2. **Distributed Systems and Networks**: Our team again, recommends Azure for distributed systems and networks. We recommend this because it offers best maximum up time using cloud-based email alerts, App Insight Logging and Monitoring Service.
3. **Security**: Hackers/Criminals have many ways of stealing data and potentially crippling your servers and ruining game play for customers. My recommendation is to pay close attention to your security protocols and ensure proper training for employees. One of the biggest threats to cyber security is lack of training and the employees themselves. Phishing and whaling are huge problems and good ways for hackers to get information from the company. Ensure proper training (recommended every 6 months and for every new hire), purchase a service such as Aura to protect information and have a dedicated monitoring team to ensure that activity is normal within the servers.