

Win 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 |
| --- | --- | --- | --- |
| 3.0 | 07/21/2023 | Mason Brown | Final Revision |

**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 Game Room would like to make a web-based game that runs on multiple platforms. The game they currently have that is available for android is called “Draw It or Lose It”. The game takes multiple teams of individuals, going four rounds at 1 minute each round. When a picture is drawn on team guesses until time runs out. If the team does not answer correctly the other teams have 15 seconds to guess the correct answer

## Requirements

*The requirements for this program include:*

* *1 or more teams*
* *Multiple people per team*
* *Team names must be unique and allow individuals to check if the name is free to use.*
* *One instance of the Game at any given moment*
* *Run successfully on multiple Platforms.*

## [Design Constraints](#_2et92p0)

*The requirements listed above is a guide to follow when creating the code for this software. The Good part about this is that we already have the game for android devices. However, we need to adapt it to the other Platforms which includes possible coding differences. The best way to approach this is to come up with a way to integrate or rewrite the code to fit Apple’s Swift coding or allow the code to inherit other coding languages. This decision should be discussed to find out the best outcome for the software.*

## [System Architecture View](#_ilbxbyevv6b6)

To be Determined

## [Domain Model](#_8h2ehzxfam4o)

The UML class diagram below shows the relationship between Entity with Game, Team, and Player class. This indicates that all three will inherit some form of information from Entity. The information shared between the three will be very similar and include things like name and id. Given this we can identify Entity as a Super Class. The classes also share an instance with another class. This is called Aggregation. In the chart this is shown with GameService having a reference of Games, Games with Team and Team with 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** | Terminal commands that allow flexibility to configure the server, access, or make changes.  Mac is popular in web hosting.  Advantages include upgradability and it has Multiple options for different web hosting requirements.  Disadvantages include It being less preferred for web hosting services | The same goes for mac plus more cost friendly.  Linux is Secured, and most preferred.  Advantages include Security flaws are caught sooner to prevent larger issues and it is the most preferred choice for web hosting services.  Disadvantages include it being more difficult to find applications to support the web hosting required needs. | More software available compared to other OS.  Windows is the dominant platform.  Advantages includes High resource requirements, less loading time, and high comfortability.  Disadvantages include virus susceptibility and poor tech support. | Considered to be better when the server is immobile and can be tracked. Specifications are better in other devices.  Mobile Devices are highly portable and more popular.  Advantages include Having a wider reach, better compatibility, and very cost-effective.  Disadvantages include being highly selective to various mobile devices and Poor security. |
| **Client Side** | Moderate expertise and time required. Cost similar to windows. | High expertise and time required. Cost is Minimal | Minimum expertise and time required. The cost is similar to mac. | Provides flexibility to clients to see updates. Slightly more difficult to implement than other devices. Cost Is considered Moderate but is dependent on Android or IOS or both combined. |
| **Development Tools** | Languages:  Mac does support most if not all language libraries. However, the implementation of languages outside Apples Swift language is limited at certain points.  Coding Software Programs:   * Swift * Notepad++ | Languages:  Linux supports Most Coding Languages however C++, C and Python are the most frequently used.  Coding Software Programs:   * Visual Studio * Eclipse * Notepad++ | Languages:  Windows supports Most coding languages. The top languages used include C++, C and python.  Coding Software Programs:   * Visual Studio * Eclipse * Notepad++ | Languages:  You can create countless apps using android and swift. Both languages and software can be run on all three machines.  Coding Software Programs:  There are a large multitude of Application that can allow you to create and implement code.  Android Studio Program languages:   * Java * Kotlin   Ios XCode program languages:   * Swift |

## 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’d recommend The Gaming Room starts with windows devices as it has more software availability. Along with that a minimum expertise and cost to get this project started. Also, you won’t run into a shortage when it comes to IDE’s.
2. **Operating Systems Architectures**: Windows offers a wide range of essential services for all Windows-based programs, empowering them to display a Graphical User Interface (GUI) while seamlessly accessing system resources and beyond. These applications also utilize Graphics and Multimedia functionalities, along with messaging and web services. These versatile services are accessible through either a user account or a dedicated server. This allows for a smooth and efficient experience for users, whether they are running applications locally on their own machines or accessing them through a centralized server. The combination of these services ensures that Windows-based applications can effectively interact with users, handle various types of media, and communicate with other systems over the internet.
3. **Storage Management**: Windows OS offers a robust storage management system to efficiently handle data and optimize storage resources. Users can use File Explorer to navigate, organize, and manage files and folders. Disk Cleanup helps free up disk space by removing unnecessary files. Disk Defragmenter or Optimize Drives reorganizes fragmented data for improved system performance. Storage Spaces allows users to create virtual drives across multiple disks, ensuring flexible and resilient storage management.
4. **Memory Management**: Windows OS incorporates a powerful memory management system that efficiently allocates and deallocates memory resources to running processes and applications. It utilizes Virtual Memory, combining RAM and a portion of the hard drive, to create the illusion of more memory space. This ensures smooth application performance even with limited physical RAM. Memory optimization techniques, like caching frequently accessed data, enhance overall system performance. Additionally, automatic memory page management, swapping data between RAM and the disk, prioritizes active processes and maintains system stability.
5. **Distributed Systems and Networks**: Windows OS utilizes sophisticated distributed systems and networks, enabling seamless communication and resource sharing among devices. It employs various networking protocols for reliable data transmission and supports centralized management through Active Directory. Additionally, peer-to-peer networking and cloud-based services enhance resource sharing and accessibility.
6. **Security**: Windows OS prioritizes security with built-in features like Windows Defender for real-time antivirus and antimalware protection, Windows Firewall for network security, and User Account Control (UAC) to prevent unauthorized access. Regular security updates and patches further ensure the system's resilience against emerging threats, making Windows OS a secure choice for users.

**Briefly summarize The Gaming Room client and their software requirements. Who was the client? What type of software did they want you to design?** The Gaming Room's client was a gaming community looking to create a software solution to enhance their user experience and engagement. They wanted to design a custom application that would serve as a centralized hub for gamers that provided features like real-time chat, game matchmaking, user profiles, leaderboards, and forums. The goal was to create a comprehensive platform that would foster interaction, competition, and communication among their gaming community members.

**What did you do particularly well in developing this documentation?** I believe what I did well in developing this documentation is I made sure to be as detailed as possible while also being short and to the point of what would make this software cost effective and give the users the highest quality end product.

**What about the process of working through a design document did you find helpful when developing the code?** When it comes to this working process the section that’s the most helpful when it comes to writing code is the requirements needed for the end product. This gives the developers a clear direction in what coding language is the best option. It also gives clear directions on what the code needs to do for the end product.

**If you could choose one part of your work on these documents to revise, what would you pick? How would you improve it?** If I were to revise one part of my work I would look over the Evaluation section. I would make it more clear and more detailed to fully show the value of each OS to help with creating a Clearer picture of the Pros and Cons of each OS for each section.

**How did you interpret the user’s needs and implement them into your software design? Why is it so important to consider the user’s needs when designing?** User needs are Vital to any type of software. Without incorporating user needs any software won’t meet the standards of its users and more often than not, they will be unsuccessful products. When you include User needs when designing you get a product that users are happy to use and dramatically improve the success of the final product. Another reason why user needs are important is for after the release of the product and attending to bugs when users point them out.

**How did you approach designing software? What techniques or strategies would you use in the future to analyze and design a similar software application?** When designing software like The Gaming Room's application, I follow a user-focused process. I create mockups for user interface design. I use an agile approach to break the project into iterations. After I choose the tech for scalability, design a modular system, manage data efficiently in databases, ensure security, conduct thorough testing, and automate deployment. For future projects, I'd adapt these strategies based on specific needs, focusing on user needs, flexibility, and iterative improvements.