

<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 | <03/18/2024> | <Domenick Dobbs> | <Final Submission> |

**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)

This "Draw It or Lose It" game is a web-based application where teams compete to guess drawings generated from a library of predefined stock images. The game has 4 rounds with drawings displayed one at a time. The software design must address the need for a scalable, distributed gaming environment that supports multiple teams and unique game and team names. The solution utilizes a singleton pattern for managing game data.

## 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)

The game, "Draw It or Lose It", is limited by the need for a web-based distributed environment, which requires thought into scalability, network latency, and data synchronization. The requirement for a unique game and team names will require a robust system to handle the name checking and validation. The singleton pattern is what is used for the game service. It has its own constraints on the management of the game data.

## [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- A base class for common attributes and behaviors – newly added.

Game- Represents a game instance, containing teams and managing game state and game loop.

Team- Represents a team within a game, containing players and a team name.

Player- Represents an individual player within a team.

**"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 good when it comes to stability and security, making it an excellent choice for hosting web applications. It may not be as cost-effective as Linux due to licensing fees. | Linux is widely used for server-side hosting due to its reliability, flexibility, and open-source nature, making it a cost-effective option. | Windows is suitable for hosting web applications, especially if they are built using .NET technologies. Sadly, it will likely require more resources and licensing costs compared to Linux. | Mobile devices are not normally used for server-side hosting due to their limited resources and processing power. |
| **Client Side** | Mac supports browsers and development tools, making it an ideal platform for client-side development. It may require more investment in terms of cost and expertise though. | Linux supports different browsers and development tools, making it a suitable platform for client-side development. It will require more technical expertise. | Windows supports a wide range of browsers and development tools, making it a popular platform for client-side development. It will likely require more investment in terms of cost and expertise. | Mobile devices require specific considerations for client-side development, such as responsive design and touch interactions. Development may be more time-consuming and require expertise in mobile-specific technologies. |
| **Development Tools** | Mac supports various programming languages and tools, including Xcode for Swift and Objective-C, as well as IDEs like Visual Studio Code and JetBrains IntelliJ IDEA for web development. (Had to look this one up) | Linux supports a wide range of programming languages and tools, including GCC for C/C++, Python, Ruby, and IDEs like Eclipse and Visual Studio Code. | Windows supports various programming languages and tools, including Visual Studio for .NET languages, as well as other IDEs like Visual Studio Code and JetBrains IntelliJ IDEA for web development. | Mobile development requires specific tools, such as Android Studio for Android development and Xcode for iOS development. Cross-platform tools like React Native and Flutter are also commonly used. |

## 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 recommend a platform like Linux for its reliability and performance in server-side hosting.
2. **Operating Systems Architectures**: The chosen operating platform for the "Draw It or Lose It" game is Linux, which is based on the Linux kernel architecture. This architecture is known for its modularity, allowing for customization and optimization for specific use cases. I still need to learn more about it though.
3. **Storage Management**: MySQL or PostgreSQL. Both are open-source relational database management systems that are widely used in web applications for their reliability, performance, and support for complex data structures. We can use this to retrieve game data and game info efficiently.
4. **Memory Management**: Linux uses different techniques for memory management. Virtual memory, paging, and swapping are just a few. Linux can efficiently allocate and manage memory resources to ensure smooth game performance.
5. **Distributed Systems and Networks**: To enable the game to communicate between various platforms, a distributed software architecture can be employed, using technologies such as RESTful APIs for server-client communication and WebSocket for real-time data exchange.
6. **Security**: On the Linux platform, security measures can include using SSL/TLS for secure data transmission, implementing authentication and authorization mechanisms to control access, and using encryption for sensitive data storage.