

Draw 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 |
| --- | --- | --- | --- |
| 1.0 | 11/15/2024 | Charity Deel | Updated software design document including executive summary, technical requirements, domain model, and design constraints. |
| 2.0 | 11/29/2024 | Charity Deel | Updated software design document including Evaluation table on Server, Client and Development Tools. |
|  |  |  |  |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room has developed an exciting game concept, Draw It or Lose It, inspired by the classic TV show “Win, Lose or Draw”. The game currently runs exclusively on Android but is now expanding to a web-based, multi-platform environment. This plan outlines a comprehensive software design solution that ensures scalability, efficiency, and security while maintaining a seamless user experience. Our approach considers team-based gameplay mechanics, unique identifiers for player and game sessions, and constraints of a web-based distribution system. The proposed system will enable the creation of multiple teams, each with several players, and ensure unique identifiers for games and players to prevent session conflicts. The design will emphasize performance optimization and platform independence to engage a broader audience.

## Requirements

* A game will have the ability to have one or more teams involved.
* Each team will have multiple players assigned to it.
* Game and team names must be unique to allow users to check whether a name is in use when choosing a team name.
* Only one instance of the game can exist in memory at any given time. This can be accomplished by creating unique identifiers for each instance of a game, team, or player.

## [Design Constraints](#_2et92p0)

* Platform Independence: The game should be compatible across various platforms, including different web browsers and mobile devices.
* Data Consistency and Uniqueness: The game should have unique identifiers for games and players.
* Scalability and Performance: The system should handle simultaneous game sessions and multiple players without compromising performance.
* Security and Data Protection: The game should implement security measures like data encryption, secure login processes, and attack protection since it involves user data.
* Session Management: Only one instance of the game should be in memory at any given time.

## [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)

The diagram represents the core entities and their relationships. The main classes are Game, Team, Player, and SessionManager. Game manages the game logic which includes the timer, library of images, and rules for guessing. It interacts with Team objects to keep track of the teams involved. Team manages the group of players. Each team object will have a unique team name and a list of player objects. Methods will manage the team activities and make sure the game rules are followed. Player holds the information about individual players like their name, score, and team assignment. SessionManager handles the creation, uniqueness, and management of game sessions. Encapsulation is shown by each class managing its data and behavior. Player class encapsulates player details, and Game class manages the games state and logic. Inheritance is shown with the Entity class acting as a mother case for Player and Team classes. Polymorphism is shown with the Game class using polymorphic methods to render images differently based on game stage. Abstraction is shown with complex game logic being simplified into methods within the Game class, making it easier to manage and maintain.

**"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 offers robust security features, a stable environment for web applications, and excellent developer tools like Xcode. However, hosting on macOS is less common and can be cost-prohibitive due to licensing and hardware requirements. Scalability options are limited compared to other platforms like Linux. | Linux is an ideal choice for hosting due to its scalability, open-source nature, and flexibility. It offers strong performance for server-based applications and robust support for web frameworks like Apache and Nginx. The main drawback is the steep learning curve for those unfamiliar with its ecosystem. | Windows Server provides excellent compatibility with Microsoft tools, such as .NET and SQL Server, which can streamline development. However, the licensing costs can be high, and Windows tends to require more frequent updates and maintenance than Linux. | While mobile devices are primarily client-side, their hosting capabilities like personal servers or PWA distribution are minimal. Ensuring compatibility with mobile platforms focuses more on client-side optimizations like responsive design and API connectivity. |
| **Client Side** | Developing for macOS requires expertise in Swift or Objective-C, alongside access to macOS hardware for testing. Development costs include licensing for macOS and associated tools. | Linux development can be more affordable due to its open-source nature. However, it requires expertise in tools like Eclipse or command-line development. Testing across different Linux distributions can also be challenging. | Windows development benefits from tools like Visual Studio, which simplify integration with .NET. Expertise in C#, C++, or similar languages is necessary, alongside licenses for development tools. | Development for mobile platforms like iOS and Android requires tools such as Xcode and Android Studio. Cross-platform frameworks like Flutter or React Native can reduce development time and costs but require expertise in JavaScript or Dart. |
| **Development Tools** | Xcode is the primary IDE, supported by languages like Swift and Objective-C. Tools for testing include macOS virtual machines or physical devices. | IDEs like Eclipse or Visual Studio Code are commonly used, alongside languages like Python, Java, and C++. Frameworks such as Django or Flask offer powerful options for web development. | Visual Studio is the go-to IDE for Windows development, with support for languages like C#, .NET, and C++. It also integrates well with Azure for hosting. | Android Studio (for Android) and Xcode (for iOS) are required. For cross-platform development, React Native, Flutter, or Xamarin are common choices. |

## 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**: <Recommend an appropriate operating platform that will allow The Gaming Room to expand Draw It or Lose It to other computing environments.>
2. **Operating Systems Architectures**: <Describe the details of the chosen operating platform architectures.>
3. **Storage Management**: <Identify an appropriate storage management system to be used with the recommended operating platform.>
4. **Memory Management**: <Explain how the recommended operating platform uses memory management techniques for the Draw It or Lose It software.>
5. **Distributed Systems and Networks**: <Knowing that the client would like Draw It or Lose It to communicate between various platforms, explain how this may be accomplished with distributed software and the network that connects the devices. Consider the dependencies between the components within the distributed systems and networks (connectivity, outages, and so on).>
6. **Security**: <Security is a must-have for the client. Explain how to protect user information on and between various platforms. Consider the user protection and security capabilities of the recommended operating platform.>