

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.2

## 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 | 06/02/2024 | Brandon Brown | First revision |
| 1.1 | 06/16/2024 | Brandon Brown | Added elements to the table below evaluation |

**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 has requested the development of a game based on the famous 1980s television show Win, Lose, or Draw. “Draw It or Lose It” will be a web-based game where players will guess a phrase, title, or thing, using rendered images as clues. Games will consist of four rounds of up to a minute. Teams will have that minute to guess and if time runs out opposing teams will have 15 seconds to solve the puzzle. Images will be rendered at a steady rate and be completed in 30 seconds.

## Requirements

* *One or more teams per game are required.*
* *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.*
* *One instance of the game can exist at any time.*

## [Design Constraints](#_2et92p0)

* Web-based Distributed Environment: Draw It or Lose It must be developed through a web-based platform. Network connectivity, device compatibility, and security issues will present constraints.
  + Security is imperative, especially in a web-based environment. Protection against vulnerabilities concerning multiple platforms is important.
* The game servers need to be able to handle a large number of players without causing slowdowns or crashing.

## [System Architecture View](#_ilbxbyevv6b6)

## [Domain Model](#_8h2ehzxfam4o)

<Describe the UML class diagram provided below. Explain how the classes relate to each other. Identify any object-oriented programming principles that are demonstrated in the diagram and how they are used to fulfill the software requirements efficiently.>

**"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.**

**The Entity class is the superclass of this application and contains the id and name attributes shared by all other classes. The Game, Team, and Player classes would all inherit from the Entity superclass. We see the aggregation between the Team and Player classes and the Game and Team classes. This shows us that Teams consist of many Players, and Games consist of many Teams. GameService holds many methods the application requires to function. The ProgramDriver class is where the main() function is stored. This starts an instance of the SingletonTester to test the code.**

## [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** | Ease of use and accessibility. Great GUI. Secure environment. Less commonly used. More expensive. Not optimized for high-demand server tasks. | Cost efficient. Strong and secure environment. High performance. Can be difficult to navigate. Less likely to be compatible. | User-friendly and very compatible. Strong support from Microsoft. Can be expensive. Security vulnerabilities. | Can manage everything on the go. Devices can vary from user to user. Potential security risks |
| **Client Side** | Expensive hardware and software costs. Learning curve for developers who aren’t familiar with the system. | Cost efficient as there are minimal to no costs for deployment. The steep learning curve for those unfamiliar with the CLI. Must have a strong understanding of Linux and scripting languages. | Expensive when using Microsoft server products. Slight learning curve for developers who are unfamiliar with Windows. | Costs are dependent on Apple or Google Play store. Difficulty varies on the OS used. iOS has more specific requirements and tools. Slower development for cross-platform compatibility. |
| **Development Tools** | Xcode, VS Code, Swift, possibly UNIX. C variants, Java, PHP | SQL, Apache, Python, C variants, Java, VS Code, Eclipse, PyCharm | .NET Framework, Visual Studio, VS Code, C variants, Python, JS, HTML, Eclipse, PyCharm | Android Studio, Xcode, Visual Studio, Flutter, AppCode, Eclipse |

## 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**: Windows is the recommended operating system as there are higher compatibility and user-friendly options when using this OS. The costs aren’t overly expensive and there are many programming languages and IDEs that are available through Windows.
2. **Operating Systems Architectures**: Windows is a graphical OS created by Microsoft. It offers high performance because of its monolithic kernel architecture. It is very flexible and can support various applications and hardware configurations.
3. **Storage Management**: Microsoft Azure offers cloud storage that is highly available, scalable, and durable. It provides advanced security and compliance solutions. It supports a broad range of programming languages, frameworks, and databases and implements strong security features. With a “pay as you go” price model, businesses can manage the cost.
4. **Memory Management**: Windows can track memory location efficiently as RAM is allocated whenever a program is opened, freeing up memory. It can handle large data sets and maintain stable performance. It can manage complex drawings and user interactions effectively.
5. **Distributed Systems and Networks**: By adopting a distributed software architecture and leveraging network connectivity, Draw It or Lose It can communicate seamlessly between various platforms. Client-server models can be implemented to run the application on multiple platforms and communicate with a central server.
6. **Security**: Several measures will be implemented for security. Data Encryption, Secure Authentication, and secure networks can protect users on various platforms, ensuring data confidentiality and integrity.