

# **Draw It or Lose It**

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**Draw It or Lose It\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** 1](#_Toc145838747)

[**CS 230 Project Software Design Template\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** 1](#_Toc145838748)

[Table of Contents**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** 2](#_Toc145838749)

[Document Revision History**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** 2](#_Toc145838750)

[Executive Summary**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** 3](#_Toc145838751)

[Requirements**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** 3](#_Toc145838752)

[Design Constraints**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** 3](#_Toc145838753)

[System Architecture View**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** 3](#_Toc145838754)

[Domain Model**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** 3](#_Toc145838755)

UML Diagram**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** 4

[Evaluation**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** 5](#_Toc145838756)

[Recommendations**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** 6](#_Toc145838757)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 09/17/23 | Talon LeFevre | Initial version of the software design document for the Draw It or Lose It game. |
| 2.0 | 9/29/23 | Talon LeFevre | Filled out the Evaluation Table with findings |
| 3.0 | 10/13/23 | Talon LeFevre | Recommending Linux |

## [Executive Summary](#_sbfa50wo7nsh)

The challenge lies in expanding the currently Android-exclusive game, "Draw It or Lose It," to a broader, web-based environment suitable for various platforms. Our solution proposes a unified, scalable software design that accommodates multi-team gameplay. This approach allows players from different devices to compete seamlessly, leveraging a vast library of stock drawings to enhance the game experience.

## Requirements

The software is expected to:

Support multiple teams within a single game.

Allow each team to have numerous players.

Ensure unique games and team names.

Retain just one instance of the game in memory at any given time using unique identifiers.

## [Design Constraints](#_2et92p0)

Developing a web-based game entails constraint like:

Performance: Web applications can sometimes lag compared to native apps.

Cross-platform Compatibility: Ensuring smooth gameplay across different browsers and devices.

Latency: Real-time games require minimal latency.

Scalability: Must support a growing number of simultaneous players.

These constraints imply careful selection of a hosting environment, optimization strategies, and rigorous testing across platforms.

## [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 UML class diagram showcases the core entities: Game, Team, Player, along with the support of GameService to handle game operations. The Entity class acts as a base, embodying the attributes shared across Game, Team, and Player. The relationships indicate that one GameService can manage multiple games, each game can have multiple teams, and each team can comprise multiple players. Object-Oriented principles, such as inheritance (from Entity class) and encapsulation (methods and attributes within classes), ensure a streamlined, maintainable software design.

**"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 must work together.

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Mac OS offers a stable environment, but it's not popularly chosen for hosting web-based applications due to cost constraints. | Highly popular for web hosting due to its stability, security, and open-source nature, making it cost-effective. | Offers Windows Server tailored for web hosting with integrated .NET support but might be costlier than Linux. | Not ideal for hosting due to limited resources and potential connectivity issues. |
| **Client Side** | Development for Mac browsers entails good aesthetic designs, compatibility checks especially for Safari. | Linux users may use a variety of browsers; hence compatibility testing is essential. | Development must ensure compatibility with browsers like Edge and Internet Explorer. | Mobile-first design, responsive layouts, and touch optimization are crucial. |
| **Development Tools** | Swift, Xcode. | Python, Java, C++, and IDEs like Eclipse or JetBrains. | C#, .NET, Visual Studio. | Java (Android), Swift (iOS), React Native (Cross-platform). |

## 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**: Given the cost-effectiveness and versatility, Linux is recommended for hosting the game.
2. **Operating Systems Architectures**: Linux provides a monolithic kernel, offering faster performance and robustness.
3. **Storage Management**: PostgreSQL or MongoDB can be adopted, offering scalability and speed.
4. **Memory Management**: Linux uses virtual memory, paging, and swap space efficiently, ensuring smooth gameplay.
5. **Distributed Systems and Networks**: Utilizing cloud platforms like AWS or Azure can help distribute the game effectively, with load balancers and Content Delivery Networks (CDNs) to ensure fast, global access.
6. **Security**: Implement SSL for encrypted connections, utilize platforms offering DDoS protection, and follow best practices for database security.