

Draw It or Lose It Web-based Software Application Game

# **CS 230 Project Software Design**

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 | 05 November 2023 | Brianna De La Riva | This document is the software design for the Draw It or Lose It web based software application game for the client The Gaming Room. This is the first version of the document and discusses key requirements to incorporate into the design and any design constraints to keep in mind. |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room has indicated that they need to create an online version of their present Android app, "Draw It or Lose It." In this game, players must produce pictures from a sizable stock drawing collection to serve as hints for several teams. Play is divided into four rounds, each lasting one minute in each game. Predicting the drawing before the time restriction expires is the key objective. Other teams get a 15-second guess if a team is unable to solve the challenge in the first 30 seconds.

We suggest creating a web-based gaming application that complies with the following specifications:

1. One or more teams will be supported by the game.

2. A team may be assigned with more than one person.

3. To prevent naming conflicts, team and game names need to be distinct.

4. The game can only have one instance open in memory at once. To make sure this restriction is fulfilled, distinct identifiers will be generated for every instance of a game, team, or player.

Our suggested solution is creating a web application utilizing contemporary web technologies in order to meet these needs. This program will give users a simple interface via which they may play the game, join teams, and interact with the illustrations. A backend framework for managing players, teams, and game instances will also be included. In order to maintain data integrity and avoid memory overloads, each game, team, and player will have their own unique identity.

## Requirements

1. One or more teams will be supported by the game.

This would fall under a technical requirement as it is necessary to develop one or more teams to be compatible with each other during gameplay.

1. A team may be assigned with more than one person.

This may be a technical requirement as it must be programmed to allow teams more than one person assigned, however it may become a business need as it may require an additional employee to help in testing.

1. To prevent naming conflicts, team and game names need to be distinct.

This is a technical requirement to keep in mind as it requires each name to be specifically unique, however it can also be a business requirement as it allows for uniqueness in the way it can be implemented.

1. The game can only have one instance open in memory at once because a singleton pattern was implemented for the game service. Distinct identifiers will be generated for every instance of a game, team, or player. Technical requirement because you may lose track of data if you had multiple instances open at once.

## [Design Constraints](#_2et92p0)

1. A Distributed Web-Based Environment: Since the game is meant to operate in a distributed web context, it should be playable on a variety of internet-connected platforms. This limitation affects the technologies and architectures selected because the game needs to be scalable, responsive, and cross-browser compatible. However, if successful, it will be able to expand “Draw It or Lose It” compatibilities.

2. In-Game Real-Time: Real-time interactions and artwork images rendering are features of the game. Due to this limitation, real-time communication protocols must be implemented to guarantee low latency and coordinated gaming for every player for every round.

3. Unique Team and Game Names: The system needs a way to instantaneously validate and reserve unique names so that users can see which game and team names are available. The design of the team’s creation and user registration procedures is impacted by this limitation.

4. One-Time Game Example: A strong management system for game instances must be created because there can only be one instance of the game running in memory at any given moment. This entails managing game state, memory utilization, and effective cleanup of finished games in addition to generating unique IDs for every game, team, and player.

7. Compatibility Across Platforms: The game application should be made to function flawlessly across a range of web browsers, operating systems, and devices to support many platforms. The selection of web technologies and responsive design techniques are influenced by this limitation to help ensure it is widely accepted.

The development of the "Draw It or Lose It" web game for The Gaming Room will need to follow these design constraints in order to be successful.

## [System Architecture View](#_ilbxbyevv6b6)

 This section is intentionally left blank per template stating no details were required.

## [Domain Model](#_8h2ehzxfam4o)

The UML class diagram below showcases boxes with titles of classes at the top. The boxes are connected through lines and formatted to display class hierarchy. There are two boxes within each class box, where the top box is a list of attributes, and the bottom is a list of methods. At the top of the hierarchy as the parent is the Entity class which has two child classes of Game, Team, and Player. Each child has a relationship with one another and is connected by a zero or more relationship.

**"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)

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | **Advantage**: Popularity for security and stability which can help in reliability and protecting data within the game.  **Disadvantage**: Limited software compatibility which may lead to application compatibility issues  **Revised Mod 5:**  An advantage of the Mac is that it is common for its security and stability which can be useful for maintaining multiple users.  A disadvantage for this with the game would be its limited software compatibility | **Advantage**: Linux runs on an open source and is very customizable. This can help make hosting web based software applications with ease if they have specific requirements.  **Disadvantage**: Linux may be challenging to learn and become familiar with, this can potentially lead to longer training and less user-friendly feel.  **Revised Mod 5:**  This is an open-source hosting which is free and is customizable. However a disadvantage would be a potentially challenging learning curve for this system . | **Advantage**: Windows is commonly known for hosting a series of web based software applications which rely heavily on Microsoft technologies like .NET, ASP.NET, and more which can lead to an easy transition.  **Disadvantage**: Windows commonly has higher licensing costs compared to the open source alternatives like Linux.  **Revised Mod 5:**  This is advantageous for the game because it is common for web-based applications, however the use of this would require higher licensing costs. | **Advantage**: Allows for flexibility in accessing and managing servers remotely.  **Disadvantage**: mobile devices are not as powerful as dedicated server management tools.  **Revised Mod 5:**  Flexibility in remote server management and directly succeeds in building a mobile app as desired. However, a disadvantage is that it can have a limited processing power. |
| **Client Side** | **Advantage:**  Mac automatically comes with a user-friendly interface and is well designed. It can be an advantage when creating access to web based software applications.  **Disadvantage**: Macs have limited compatibility with certain legacy or windows specific software  **Revised Mod 5:**  An advantage is that Mac has a very user-friendly interface and links between clouds to help with storage.  Disadvantage is still its limited compatibility with software that can be problematic for some users. | **Advantage**: Linux is very customizable and secure which can be advantageous for users who wish to tune their clients environment within the web based application.  **Disadvantage**: Linux may have limited compatibility with certain software, this may lead to alternative solutions when dealing with certain applications.  **Revised Mod 5:**  Linux allows for a customizable and secure environment, however a drawback to this is how it is compatible with a limited amount of software. | **Advantage**: Windows has a large range of compatibility with web based applications and can typically make running them a breeze.  **Disadvantage**: Windows may be more vulnerable to threats of security compared to other operating systems and requires frequent maintenance and security.  **Revised Mod 5:**  Windows allows for broad compatibility, however it can unfortunately face some security vulnerabilities and needs frequent maintenance. | **Advantage**: Offers enhanced portability of device and geographic usability of software application.  **Disadvantage**: Mobile devices have smaller screens and may limit user functionality or cause it to be limited.  **Revised Mod 5:**  Enhanced portability for user to use anytime, however a limitation is the decreased screen size and potential functionality limitations. |
| **Development Tools** | **Advantage**: Macs support a wide range of programming languages and have IDE environments like Xcode. This causes a development system.  **Disadvantage**: Macs may have limited support for IDE’s that can be more paired with Linux or Windows. This may lead to compatibility issues  **Revised Mod 5:**  An advantage is that Mac supports a variety of programming languages, Xcode. However the disadvantage with this is that there is a limited amount of IDE’s available to use | **Advantage**: Linux is highly flexible and capable of supporting various programming languages and IDE’s. This makes it a logical choice for developers where open-source capabilities and customizations is needed.  **Disadvantage**: Some Windows specific tools and software may not be usable for Linux. This can potentially lead to a delay in developing the software and determining alternatives.  **Revised Mod 5:**  There is a variety of programming languages accepted and IDE’s. However, there is some windows specific tools that may not be able to be used and can be problematic to some. | **Advantage**: Windows supports a large variety of programming languages and popular IDE’s like Visual Studio. This can help simplify the development process.  **Disadvantage**: Developing only on windows can lead to compatibility issues or require additional testing to ensure best execution.  **Revised Mod 5:**  A positive to this is that it is able to support various languages and use IDE like Visual studio. However there can be some compatibility difficulties that may need more testings. | **Advantage**: Mobile devices can support web development through text editors or apps. It can also allow for portable and readily available tools since it is in hand.  **Disadvantage**: The smaller screen size and limited processing power of mobile devices can cause difficulty when tackling complex problems.  **Revised Mod 5:**  It allows for web development through apps or editors however a disadvantage again may be the smaller screen and decreased processing power. |

## Recommendations

1. **Operating Platform**: I recommend building the Draw It or Lose It system on top of a cross-platform operating system, like Linux. With a dynamic user base, Linux is very adaptable and open-source.

**Revised Mod 7:** Linux offers the Draw It or Lose It system a stable and adaptable base. Because of its modular architecture, which guarantees scalability and flexibility, it may be used in a variety of computer settings.

1. **Operating Systems Architectures**: Linux is built upon a modular architecture, this allows for ease of flexibility and scalability. This operating system can be suitable for various computing environments

**Revised Mod 7:**

The modular architecture of Linux has the benefit of scalability and flexibility. Because of its structure, Draw It or Lose It performs at its best even in a variety of computing conditions.

1. **Storage Management**: The Draw It or Lose It web based game would likely have a combination of traditional files as well as cloud based storage. This can include potentially using a Google Cloud Storage, or Amazon S3

**Revised Mod 7:** For Draw It or Lose It, a mixed strategy is advised for storage management. The integration of cloud-based storage solutions such as Amazon S3 or Google Cloud Storage with traditional files guarantees scalability, accessibility, and effective data processing.

1. **Memory Management**: Linux utilizes a virtual memory management strategy to manage its memory resources. This can include things like swap space to optimize memory usage or demand paging.

**Revised Mod 7:**

Draw It or Lose It heavily relies on Linux's virtual memory management technique. To optimize memory utilization and ensure smooth and effective game performance, strategies like demand paging and swap space will be used.

1. **Distributed Systems and Networks**: A distributed system architecture should be considered if communication between various platforms can be achieved. This can help improve real-time interactions within the game. This can be implemented by using RESTful API’s or WebSocket. Additionally, it would be wise to consider potential connectivity or outage issues in the future and ensure there is enough initial load balances.

**Revised Mod 7:**

For Draw It or Lose It, a distributed system design is advised in order to provide smooth cross-platform communication. Using WebSocket or RESTful APIs will improve real-time communication. Furthermore, it is imperative to take into account possible problems with connectivity or outages, which means that initial load balances must be included in adequate amounts to ensure a dependable gaming experience.

1. **Security Mod 7 Addition:**

Draw It or Lose It places a high priority on security. We'll use Linux's strong security features to protect user data on multiple systems. Because Linux is open-source, it is constantly inspected and updated, improving the gaming system's overall security posture. To ensure a safe gaming environment, frequent security assessments and fixes will be put into place to handle new threats and vulnerabilities.