

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.1

## 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 |
| --- | --- | --- | --- |
| 1.2 | 02/20/25 | Matthew Dunfee | Recommendations |

**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 wants to extend their domain for the pre-existing game Draw It or Lose It. It is currently available in an Android app only. Their goal is to broaden their audience by serving multiple platforms via a web-based application.

## Requirements

* Must be Web a based application
* Must have time-based rounds
* Must support multiplayer
* Must build teams

## [Design Constraints](#_2et92p0)

* Must support multiple browsers
* Must utilize the Singleton pattern
* Must adhere to best practices

## [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 OOP concepts present in the Domain Model are Inheritance, Polymorphism, and Encapsulation.

* Inheritance is present by the indication of the arrow with a white or clear head. It shows that Game, Team, and Player are all extensions of the Entity Class.
* Polymorphism is present in ‘GameService’ as method overloading. It can be seen where getGame() has two different function definitions because its input changes from one to another despite the end result being the same. It is also present in the overriding of the toString functions.
* Encapsulations is present by the restriction of data attributes. This results in the additional notable function families being present (getters/setters).

**"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** | Pros:  Mac OS X Server offered secure and performant,  standardized env.  Cons: It was a paid service and is no longer supported as of 2022. | Pros: Linux Servers are free and open source. They are stable and secure.  Cons: They need extra effort to set up. | Pros:  Windows Server is stable and secure.  Cons: Windows Server is a paid service and a prime target for cyber threats. | Hosting on mobile is not the best practice. |
| **Client Side** | Mac uses Safari by default. This browser has different requirements than other popular Browsers but should only increase the cost for minor client-side adjustments and maintenance. | Linus uses Firefox by default. This Browser has different requirements than other popular browsers but should only increase the cost for minor client-side adjustments and maintenance. | Windows uses a Chromium-based browser by default. This Browser has different requirements than other popular browsers but should only increase the cost for minor client-side adjustments and maintenance. | IPhone uses Safari as a default browser. Android uses a Chromium-based Browser. These are already covered. |
| **Development Tools** | OS X server doesn’t exist anymore.   No additional tools are needed to support Safari. | Node, VScode, Postgres, React, supporting lib and frameworks for development. Docker, ISP, Hosting platform, monitoring software, additional DevOps.  Cost: hosting, ISP fee for a domain name. Plus, Postgres Storage. | Java, IntelliJ, Oracle, Vue, and supporting lib and frameworks for development. Docker, ISP, Hosting platform, monitoring software, additional DevOps.  Cost:  hosting, ISP fee for a domain name. Plus, Windows key, IntelliJ subscription, Oracle subscription, and Oracle storage. | Developing mobile is not the best practice. |

## 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**

The most suitable platform for this application is a web-based one. This is primarily because it significantly reduces development complexity and cost. By using a decoupled architecture, we can eliminate redundant work by reusing the server in a pinwheel architecture, effectively simplifying the system. Furthermore, by choosing a web browser-based approach instead of developing native applications for multiple platforms, we minimize platform-specific development efforts, resulting in a single client and a single server for one unified application.

1. **Operating System Architectures**

One critical piece of missing information in this project is expected user volume. We do not have data on the number of users or the expected throughput, which makes it difficult to determine precise resource requirements. Given this uncertainty, the best approach is to prioritize scalability and cost efficiency until more concrete usage data is available.

For this reason, I recommend using Linux-based Docker containers for hosting the application on remote servers. This choice provides scalability, cost efficiency, and flexibility, allowing easy deployment across cloud platforms. Additionally, since hardware concerns are abstracted away, this approach eliminates the need for dedicated infrastructure investments, keeping initial costs low.

1. **Storage Management**

Storage management is best handled within a flexible cloud infrastructure, as this approach allows deployment without requiring precise predictions of storage needs. Cloud storage solutions enable on-demand scaling, ensuring that costs remain proportional to actual usage rather than requiring upfront investment in physical hardware. Additionally, cloud storage solutions provide seamless migration options, meaning data can be transferred to on-premise infrastructure in the future without significant disruptions.

1. **Memory Management**

Memory management is particularly important in cloud-based infrastructures, as bandwidth usage directly affects operational costs. Several techniques can be used to optimize memory management and minimize data transfer fees:

* Cookies: Reduce server calls by storing session data on the client side.
* State Management: Store certain site states locally on the client side, reducing server dependencies and egress costs.
* Caching: Cache frequently used assets, such as game images, to reduce redundant data requests and lower bandwidth consumption.

By implementing these memory management techniques, network traffic can be optimized, resulting in improved performance and lower operational costs.

1. **Distributed Systems and Networks**

On the client side, additional attention must be given to browser compatibility and performance optimizations to ensure a smooth user experience. However, most backend scalability, redundancy, and networking concerns will be handled by the cloud service provider.

If the client were to opt for on-premise servers and storage, several additional considerations would come into play, including:

* Redundancy (to ensure data availability),
* Performance optimization (for faster response times), and
* Scalability (to accommodate future growth).

By leveraging cloud services, these challenges are offloaded to the provider, reducing the complexity of infrastructure management.

1. **Security**

Security concerns for this application primarily fall into two areas: network security and application security.

* Network Security:
  + API keys should be secured and properly rotated to prevent unauthorized access.
  + Encryption protocols (such as TLS) must be enforced for data in motion.
* Application Security:
  + Data at rest is vulnerable to attacks such as SQL injection and other common security threats.
  + Proper input validation and secure coding practices must be implemented to prevent breaches.
  + Critical environment variables should be stored securely and not exposed in source code.

Since most infrastructure security responsibilities will fall under the cloud provider, monitoring solutions should be implemented to ensure ongoing security compliance. While this application does not handle highly sensitive user data, it is still essential to follow best practices to protect both user information and system integrity.