

# Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 3.0

## Table of Contents

[**CS 230 Project Software Design Template 1**](#_Toc164530106)

[**Table of Contents 2**](#_Toc164530107)

[**Document Revision History 2**](#_Toc164530108)

[**Executive Summary 3**](#_Toc164530109)

[**Requirements 3**](#_Toc164530110)

[**Design Constraints 3**](#_Toc164530111)

[**System Architecture View 3**](#_Toc164530112)

[**Domain Model 3**](#_Toc164530113)

[**Evaluation 4**](#_Toc164530114)

[**Advantages & Disadvantages 5**](#_Toc164530115)

[**Recommendations 6**](#_Toc164530116)

[**References 7**](#_Toc164530117)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 03/24/24 | Mubeen Ahmed Khan | Initial Iteration. Added business and technical constraints, domain model narrative, executive summary, system architecture and recommendations. |
| 2.0 | 07/04/24 | Mubeen Ahmed Khan | Updated to reflect evaluation table update for better clarity regarding server side, client side and development tool requirements and possible limitations |
| 3.0 | 20/04/24 | Mubeen Ahmed Khan | Added advantages & disadvantages section to summarize the information from evaluation section; added updated recommendations to provide clear information on Linux and added references. |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room needs help expediting the development process to create a web-based version of their gaming application. Important software requirements are included in the suggested software design paper. Enforcing a singleton pattern to permit just one instance of a game in memory at a time, enabling numerous teams and players, and guaranteeing unique game and team names are all part of this. A logical system architecture, adherence to technical and business objectives, and consideration of design restrictions for web-based distributed settings are all part of the suggested approach.

## Requirements

* Business Requirements:
  + The capacity to play a game with several teams and players.
  + Identifiable games and team names for users.
* Technical Requirements:
  + Implement a singleton pattern to confine one instance of the game in memory.
  + For effective management of game instances and team/player collections, apply the iterator pattern.

## [Design Constraints](#_2et92p0)

* The creation of a distributed ecosystem based on the web is the main design limitation.
* Implications include:
* Ensuring scalability and performance for handling multiple concurrent users.
* Implementing robust security measures to protect user data and prevent unauthorized access.
* Adapting the user interface for various devices and screen sizes to ensure a seamless user experience.

## [System Architecture View](#_ilbxbyevv6b6)

A client-server architecture will be the hallmark of the system design, with the server hosting the gaming application and handling player, team, and game instance management. To manage user interactions and show game information, the client-side will communicate with the server. A database will be used to store game data, team information, and player details. Storage management will take place on the server-side.

## [Domain Model](#_8h2ehzxfam4o)

The Entity class provides common features and actions that are shared by the entities in the domain model, such as Game, Team, and Player. These classes' relationships illustrate object-oriented concepts like composition and inheritance, which guarantee effective software needs fulfillment.

**"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** | Mac has powerful hosting capabilities with Nginx and Apache. However, as of April 21, 2022, “Apple has discontinued the macOS Server.” | Linux is a popular choice for server installations because of its affordability, scalability, and stability. The operating system itself has no license fees, however support expenses could be different. | Although Windows Server offers an easy-to-use interface, the server operating system and related software may come with high license charges. | Web-based apps can run on mobile devices, but there are certain drawbacks, such as lower resources than on dedicated servers. Depending on the kind of device and OS version, deployment might change. |
| **Client Side** | Developers need expertise in web development frameworks such as React or Angular for desktop browsers. Purchasing Mac hardware for testing and development may be one of the costs. | Due to differences in browsers and setups, Linux-based desktop clients may need more testing before they are considered compatible with online applications. To ensure compatibility, development time may need to be extended. | It could be necessary to make changes to Windows desktop clients to support various browsers and screen sizes. Visual Studio and other development tools can help with this procedure, although there may be license fees involved. | For cross-platform compatibility, mobile devices need to use frameworks like Bootstrap and responsive design techniques. Different screen sizes, input techniques, and OS versions may need to be considered during development, which might add to expenses and development time. |
| **Development Tools** | For macOS development, programming languages like Objective-C or Swift are frequently utilized, with Xcode serving as the main IDE. Xcode licensing fees could be involved. | Linux development commonly uses languages like Python, JavaScript, or C/C++ with tools like Visual Studio Code or JetBrains IDEs, which are often free. | Languages like C#, .NET, or JavaScript are frequently used in Windows programming, and programs like Visual Studio—whose licensing fees vary based on the version—are used. | Languages like Swift (iOS), Kotlin/Java (Android), or frameworks like React Native or Flutter (for cross-platform development) are necessary for mobile development. Most IDEs, such as Android Studio for Android and Xcode for iOS, are free, however there may be fees for third-party tools and libraries. |

## Advantages & Disadvantages

* **Mac:**
  + Advantages: Powerful hosting capabilities and a large range of development tools.
  + Disadvantages: Discontinued macOS Server; possible hardware and licensing expenses.
* **Linux:**
  + Advantages: Affordability, scalability, reliability, and a vast selection of free development tools.
  + Disadvantages: Potential support costs and additional compatibility testing
* **Windows:**
  + Advantages: User-friendly UI, extensive set of programming tools.
  + Disadvantages: There may be substantial license fees, as well as modifications to browser and screen size compatibility.
* **Mobile Devices:**
  + Advantages: Cross-platform interoperability is possible, and there are free development IDEs available.
  + Disadvantages: Not appropriate for server-based roles; possible expenses for third-party tools and libraries; extra considerations for screen widths, input modalities, and operating system versions.

## Recommendations

1. **Operating Platform**: For scalability and affordability, Linux is recommended as the operating platform for expanding the “Draw It or Lose It” game to other computing environments. “Linux is a popular choice for server installations because of its affordability, scalability, and stability.” The operating system itself has no license fees, however support expenses could be different.
2. **Operating Systems Architectures:** Linux supports flexible architectures such as ARM, x86, and others, which guarantee compatibility with a wide range of hardware configurations. This flexibility allows the game to be deployed on a variety of devices, from personal computers to servers, enhancing its accessibility and reach.
3. **Storage Management**: An appropriate storage management system to be used with the recommended operating platform is a relational database management system (RDBMS) running on Linux, such as MySQL or PostgreSQL. This will allow efficient organization and retrieval of game data, team information, and player details.
4. **Memory Management:** “Linux optimizes memory utilization using virtual memory management techniques,” which guarantees the best possible performance for the gaming program. Techniques such as efficient image loading, image compression, memory recycling, and garbage collection can be effectively implemented on this platform.
5. **Distributed Systems and Networks:** “Platforms in distributed systems might communicate more easily if RESTful APIs” were used and communication protocols like HTTP were used. This allows the game to be played across various platforms, enhancing its reach and user base. Dependencies between the components within the distributed systems and networks, such as connectivity and outages, need to be considered and handled effectively. Incorporating a Content Delivery Network (CDN) can also help reduce latency, improve availability, and save bandwidth costs. By caching material closer to players, a CDN may drastically reduce latency while loading the game, resulting in a smoother and more pleasurable gaming experience.
6. **Security:** “Linux provides strong security features like firewalls, encryption, and file permissions, guaranteeing the safety of user data on a variety of systems.” Further precautions, such as the use of HTTPS encryption, can improve data transfer security even more. This ensures that user information is protected on and between various platforms.

## References

* About macOS Server 5.7.1 and later - Apple Support. (2023, November 15). Apple Support. https://support.apple.com/en-us/101601
* Chapel, J. (2021, December 10). Cloud Storage Cost Comparison: AWS vs. Azure vs. Google. Medium. https://jaychapel.medium.com/cloud-storage-cost-comparison-aws-vs-azure-vs-google-844dfff3d324
* Ghosh, S. Distributed Systems: An Algorithmic Approach. CRC Press. Linux Security. https://www.linux.com/training-tutorials/linux-security-complete-guide/
* Linux Operating System: Everything You Need To Know. https://www.linux.com/what-is-linux/
* Silberschatz, A., Galvin, P. B., & Gagne, G. (n.d.). Operating System Concepts, 8th Edition. O’Reilly Online Learning. https://learning.oreilly.com/library/view/operating-system-concepts/9780470128725/silb\_9780470128725\_oeb\_c10\_r1.html#h1
* Understanding Linux memory management. https://opensource.com/article/19/2/linux-memory-management