

<Name-of-Software-Application>

# **CS 230 Project Software Design Template**

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 | <05/28/2023> | <Julien Dorka> | Developing for the Gaming Room |

**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 is looking to expand the availability of their game, "Draw It or Lose It," by developing a web-based version that can run on multiple platforms. Currently, the game is only available on Android. "Draw It or Lose It" is a multiplayer game where multiple teams, composed of several players, compete in four rounds, each lasting one minute.

## [Design Constraints](#_2et92p0)

Each team has multiple people

Game and Team names must be unique to allow users to check whether the name is in use or free

Needs one or more teams involved

Must run on multiple platforms in order to be fully functional

## [Domain Model](#_8h2ehzxfam4o)

In this scenario, the Entity class serves as a superclass that provides common attributes such as "name" and "id" to the Game, Team, and Player classes. This inheritance relationship means that the Game, Team, and Player classes inherit or receive information from the Entity class. Additionally, the relationships between these classes can be represented as follows -> Game and Team, The relationship between Game and Team can be described as a "has-a" relationship which means that a Game object has a reference to a Team object. In UML notation, this relationship is represented as aggregation (HAS-A). GameService and Games: The GameService class has a reference to Games. This suggests that GameService manages or handles instances of the Games class. Again, in UML notation, this relationship is represented as aggregation. Team and Player: The Team class has a reference to Player objects. This indicates that a Team consists of one or more Player instances. Similarly, in UML notation, this relationship is represented as aggregation.

**"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** | Regarding hosting websites on Mac, the macOS operating system does offer a server-based deployment method. Macs can be configured as web servers using software such as Apache or Nginx, allowing for website hosting.  However, it's important to note that macOS Server, the dedicated server edition of macOS, was discontinued by Apple starting with macOS Catalina (10.15). Instead, Apple now provides the Server app, which offers a more streamlined and integrated set of server features within the standard macOS.  As for potential licensing costs, the server operating system for Macs does not have separate licensing fees. | Linux offers a server-based deployment method for hosting websites. Linux distributions are widely used for web server hosting due to their stability, security, and scalability. Commonly used web server software like Apache, Nginx, and lighttpd can be easily installed and configured on Linux servers to host websites.  When it comes to licensing costs, Linux is an open-source operating system. This means that it is generally available for free, and clients like The Gaming Room would not have to pay licensing fees for the server operating system itself. | Windows offers a server-based deployment method for hosting websites. The Windows Server operating system is specifically designed for server-based deployments and supports various web server software such as Internet Information Services (IIS).  When it comes to licensing costs for the server operating system, Windows Server does have licensing fees associated with it. The specific licensing costs depend on the edition and licensing model chosen by the client, The Gaming Room.  The licensing costs can vary based on factors such as the number of server instances, the number of users accessing the server, and the chosen licensing program. | Mobile platforms do not typically offer a server-based deployment method for hosting websites in the same way that traditional server operating systems do. Mobile platforms such as Android and iOS are primarily designed for running mobile applications rather than hosting websites directly on the device.  When it comes to hosting websites for mobile platforms, the common practice is to use a separate server infrastructure to host the website and provide APIs or web services that the mobile applications can interact with. The website is hosted on traditional server operating systems like Linux, Windows, or macOS, while the mobile applications communicate with the server via network requests. |
| **Client Side** | Ensuring compatibility with all web browser platforms and mobile devices, Mac requires following cross-platform development practices, employing responsive design, conducting thorough testing, considering Mac-specific guidelines, having the necessary expertise and resources, and accounting for additional time and potential cost implications. | Ensuring compatibility with all web browser platforms and mobile devices on Linux involves following cross-platform development practices, conducting thorough browser and mobile device testing, having expertise in Linux development, considering open-source options, and accounting for additional time and potential cost implications. | Ensuring compatibility with all web browser platforms and mobile devices on Windows involves following cross-platform development practices, conducting thorough browser and mobile device testing, having expertise in Windows development, considering Windows Store guidelines if applicable, and accounting for additional time and potential cost implications. | Ensuring compatibility with all web browser platforms and mobile devices requires utilizing cross-platform development frameworks, implementing responsive design, leveraging native features and APIs, conducting thorough testing on multiple devices, having expertise in the respective mobile platforms, following app store guidelines, and considering additional time and potential cost implications. |
| **Development Tools** | When it comes to running languages on Macs, Swift is a popular option that stands out. However, Macs are capable of running a wide range of languages, including but not limited to HTML, CSS, and JavaScript. Additionally, Macs support libraries that aid in frontend development as well as general-purpose languages like Java, Python, PHP, and Ruby.  Swift, specifically designed for macOS and iOS development, offers a seamless and efficient programming experience on Macs. While Xcode is available for free, there may be licensing costs associated with using tools like AppCode for macOS development.  Mac requires expertise in Swift/Objective-C, familiarity with Apple's frameworks, and the use of tools like Xcode or AppCode. | Developing software for Linux requires expertise in the chosen programming languages, Linux environment, and relevant tools and frameworks. The impact on a development team includes the need for specialized skills, collaboration between different team members, potential specialization, and consideration of compatibility across different Linux distributions. Licensing costs may vary depending on the chosen IDEs and tools, with many open-source options available. | Developing software for Windows requires expertise in the chosen programming languages, Windows platform, and relevant development tools. The impact on a development team includes the need for specialized skills, collaboration between different team members, potential specialization, and consideration of compatibility across different versions of Windows. Licensing costs may be applicable depending on the chosen IDEs and tools, with both free and paid options available. | When developing software for mobile devices requires expertise in the programming languages and frameworks specific to each platform (Swift for iOS, Java/Kotlin for Android). The impact on a development team includes the need for platform-specific knowledge, potential specialization, collaboration between different team members, and consideration of cross-platform development if targeting multiple platforms. Most of the relevant development tools and IDEs are available for free, with potential costs associated with additional libraries or services. |

## 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:**

Based on the requirement to expand Draw It or Lose It to other computing environments, I recommend utilizing the Windows operating platform. Windows offers extensive compatibility with a wide range of devices and computing environments, making it easier to expand the game to different platforms such as desktop computers, laptops, tablets, and even mobile devices. Windows also provides robust server capabilities that can support the backend infrastructure required for Draw It or Lose It.

1. **Operating Systems Architectures:**

Windows operating systems support multiple architectures, including x86, x64, and ARM. The x86 and x64 architectures are commonly used for traditional desktop and server environments, while ARM architecture is prevalent in mobile devices and certain server platforms. Windows supports both 32-bit and 64-bit versions, providing flexibility based on the specific hardware and software requirements of Draw It or Lose It.

1. **Storage Management**:

For storage, I would recommend leveraging Microsoft Azure due to its competitive prices, excellent customer support, and continuous updates and support. Azure offers a range of features that align well with the requirements of Draw It or Lose It. By leveraging Azure's storage capabilities, you can benefit from its cost efficiency and scalability. Cloud storage is increasingly popular due to its ability to efficiently manage storage resources, provide cost-effective solutions, and easily scale to meet changing demands. Overall Microsoft Azure is a reliable and feature rich solution for storage needs, offering competitive pricing excellent support and continuous updates to ensure a robust storage management system for Draw it or Lose it.

1. **Memory Management**:

Windows 10, the latest version of the Windows operating system, continues to enhance its memory management capabilities, resulting in faster and more efficient loading from memory. The improvements primarily leverage disc paging and demand paging techniques to effectively extend the computer's physical memory or RAM.

In Windows 10, each process can make full use of the virtual memory address space, providing ample memory resources for applications. This expanded virtual memory address space ensures that the application can efficiently manage memory and accommodate its memory requirements.

Windows 10's advanced memory management techniques, such as disc paging and demand paging, along with its expanded virtual memory address space, contribute to faster loading times and improved memory utilization for applications like Draw It or Lose It.

1. **Distributed Systems and Networks**:

To enable communication between Draw It or Lose It on various platforms, a distributed software architecture can be implemented. This can be achieved by utilizing technologies such as web services, APIs (Application Programming Interfaces), and standard network protocols like HTTP. Draw It or Lose It can expose APIs that allow different platforms to interact with the game's backend, exchanging data and commands. The network that connects the devices can be based on standard TCP/IP protocols, ensuring compatibility across different platforms.

Considering dependencies and potential connectivity issues, it is essential to design the distributed system with fault tolerance and redundancy in mind. Employing load balancers, redundant servers, and distributed databases can help mitigate connectivity outages and ensure high availability. Additionally, implementing error handling and retry mechanisms within the software can improve the resilience of the distributed system.

1. **Security**:

Windows operating systems provide a comprehensive set of security features to protect user information and ensure system integrity. To protect user information on and between various platforms.

Access Controls help utilize Windows security features such as user accounts, permissions, and group policies to control access to the game and protect user data from unauthorized access.

Encryption: Implement encryption mechanisms such as Windows BitLocker to encrypt data at rest and transport layer security (TLS/SSL) protocols to secure data in transit.

Firewalls and Intrusion Detection: Leverage Windows Firewall and intrusion detection systems to monitor and control network traffic, preventing unauthorized access and detecting potential threats.

User Authentication: Implement secure user authentication mechanisms, such as strong passwords, multi-factor authentication, or integration with external identity providers, to ensure user protection.

By leveraging the user protection and security capabilities of the Windows operating platform, The Gaming Room can ensure the confidentiality, integrity, and availability of user.