

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.2

## Table of Contents

[**CS 230 Project Software Design Template** 1](#_heading=h.gjdgxs)

[**Table of Contents 2**](#_heading=h.30j0zll)

[**Document Revision History 2**](#_heading=h.3znysh7)

[**Executive Summary 3**](#_heading=h.2et92p0)

[**Requirements 3**](#_heading=h.tyjcwt)

[**Design Constraints 3**](#_heading=h.1t3h5sf)

[**System Architecture View 3**](#_heading=h.4d34og8)

[**Domain Model 3**](#_heading=h.2s8eyo1)

[**Evaluation 4**](#_heading=h.17dp8vu)

[**Recommendations 5**](#_heading=h.26in1rg)

## [Document Revision History](#_heading=h.lnxbz9)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 5/27/2024 | Ty Wheeler | Revision one of the Draw It or Lose It software design documents. |
| 1.1 | 6/9/24 | Ty Wheeler | Additional information for evaluation and recommendations. |
| 1.2 | 6/17/24 | Ty Wheeler | Analysis of characteristics and techniques for specific system architectures, updating and completing the recommendations section. |

**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](#_heading=h.35nkun2)

The Client “The Gaming Room” wishes to develop their game “Draw It or Lose It” for several other platforms. It is currently a solely web-based game. In the game a user is put into a team; then several teams compete to guess what is being drawn by other player’s within a 30 second time frame. If the team fails to guess the puzzle before time is out then the other teams can offer one guess each to solve it within 15 seconds.

## Requirements

* Each game must have the ability to have one or more teams involved.
* Each of the teams must have several players assigned to each team.
* Team and game names must be unique so that user’s may check whether or not a name is already in use.
* Only one instance of the game can exist at any given time.
* The game must be able to run on several different platforms (Android, IOS, web-based, etc)

## [Design Constraints](#_heading=h.1ksv4uv)

* In order to prevent potential errors when user’s join games and enter their names, the program should check for uniqueness in the names of both team and game names.
* A single instance of the game service being run in memory at a time is integral to the functionality of the game operation.
* Draw It or Lose It must be compatible for other platforms while also working on it’s original web-based platform.

## [System Architecture View](#_heading=h.44sinio)

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](#_heading=h.2jxsxqh)

The domain model in the UML diagram illustrates the relationships between the Game, Team, and Player classes, all of which inherit from a common superclass called Entity. The Entity class provides shared attributes like 'id' and 'name' to these subclasses. In this system, a Game is composed of multiple Teams, and each Team consists of multiple Players.

The GameService class manages the lifecycle of Game instances and holds references to multiple Game objects, indicating a composition relationship. Similarly, the Game class manages its Teams, and the Team class manages its Players, reflecting composition relationships in UML.

The ProgramDriver class serves as the entry point for the application, where the main function resides. It creates a singleton instance of the GameService class, ensuring only one instance exists throughout the application. The ProgramDriver is responsible for adding games, teams, and players using the GameService instance and relies on the SingletonTester class for certain functionalities, as indicated by the <<uses>> dependency arrow in the UML diagram.

This UML class diagram showcases key object-oriented programming principles such as inheritance, encapsulation, and abstraction. Inheritance is evident in how the Entity superclass shares attributes and behaviors with its subclasses (Game, Team, and Player), reducing code duplication and promoting consistency. Encapsulation is demonstrated by the GameService class, which controls access to its attributes and methods, ensuring data privacy. Abstraction is represented by the classes' focus on managing essential interactions between games, teams, and players, while hiding unnecessary implementation details.

**"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](#_heading=h.z337ya)

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** | Very secure and accessible with a very user friendly GUI and easy to access terminal commands. However, high cost of both hardware and software as well as limited scalability are some cons.  1.1  Mac can be configured to be utilized as a web server with built in tools like Apache with many server functionalities. The cost of macOS server is approximately $20 which would be per machine, it’s important to take into consideration the higher cost of apple hardware. | Being open-source it allows for more flexibility with software implementation and tools used. Can be configured to be highly secure as well as a relatively cost effective solution.  1.1  Ubuntu can be configured as a web server software and Linux’s reputation for great stability and configurable security makes it a great platform. Many distributions are available for free where enterprise level solutions may be more costly, but more effective due to the lower cost entry point. | While not as cost effective as Linux, Windows is still a reliable platform with access to a command prompt for server commands. A plethora of compatible hardware and software support also makes Windows an enticing option.  1.1  Windows offers website hosting using Internet Information Services, and Windows Server is also capable of running Apache. However, these come at an additional cost, with Windows Server Essntials costing about $500, and Standard costing $970 plus the cost of client access licenses at $40 per user. | Ease of use and portability are convenient, however various hardware capabilities across different mobile devices and different software capabilities may cause complications.  1.1  Mobile devices like iOS and Android are not best designed for any sort of server deployment. Therefore, it is not recommended. |
| **Client Side** | User friendly OS design allows users to use the program with limited skills, however there is an increased cost for the end user using Mac.  1.1  Use of standard web technologies like JavaScript and HTML5 help increase compatibility and support in the web based application. Cross browser testing is also key to ensuring the same user experiences across different browsers on any Mac device.Tools like BrowserStack ($30 per month) help to be a valuable tool in this development process. | Not as user friendly and requires much more learning and expertise from the user. However, with Linux being free to utilize it does have a significantly lower cost.  1.1  Similarly to Mac, ensuring cross browser and platform compatibility eans testing across different browser applications and using universal web technologies like HTML5 to ensure cross browser compatibility. | Higher cost as far as licensing software however Windows is easier to learn and navigate, requiring less expertise.  1.1  Similar to other desktop operating systems ensuring compatibility using widely adopted web technologies ensures compatibility across different browsers especially Microsoft’s own Edge browser. Windows virtual machine software also provides another useful testing tool on top of BrowserStack for testing purposes. | Offers much more flexibility and more intuitive design. Wide range of availability makes it easily accessible however with a wide range of mobile devices available, may be difficult to implement properly across devices.  1.1  Approaching mobile requires us to take into consideration how design and layout should differ from desktop versions to be mobile friendly, while also being familiar to their desktop counterparts. A interface with touchscreen in mind and a smaller screen is key to a successful mobile software design. Optimizing performance with the same universal web technologies mentioned but testing with different tools like Google Lighthouse will ensure a smooth and user friendly mobile experience. Android Studio for Android and Xcode for iOS will be key in making the game run smooth on both platforms. When taking cross platform development into consideration we can look at frameworks like React Native. Developer accounts for apple are roughly $99 a year and $25 one time fee on Google Play. |
| **Development Tools** | Relevant programming languages consist of Javascript, CSS and HTML. Relevant IDEs used are Visual Studio Code and PyCharm.  1.1  Languages like HTML5, CSS3, Javascript and Swift can be used. IDE’s utilized would be Xcode and various Safari Developer tools for Mac OS. Multiple teams while not entirely necessary may be helpful in macOS development as expertise in different languages, such as Swift and Javascript, is required. The cost is relatively low as Xcode is free however the previously mentioned Apple Developer membership is $100 per year. | Relevant programming languages such as CSS, Java, and Python used in VS Code, Atom, VIM and Eclipse.  1.1  Languages to be used for Linux based development would be similarly HTML5, CSS3 and Javascript aswell as integration with Python and Ruby. IDE’s to be used on linux would be Visual Studio Code.A single team with expertise in Linux based web development should be sufficient. Linux being open source means no additional cost is required for development. | Relevant programming languages consist of C#, CSS, and Java to use in IDEs like Eclipse, PyCharm, and VS Code.  1.1  Windows development takes HTML5, CSS3 and Javascript into consideration again aswell as ASP.NET, which helps build free cross platform apps with its open source framework. Visual Studio aswell as Microsft Edge Developer could be used for development. Again, separate teams is not a requirement but two teams with specific expertise in general web technologies and another with ASP.NET and C# expertise could help accelerate progress. Visual studio and Windows-specific design parameters and required skills the team needs to take into consideration. Visual studio Community is free however Professional version runs $45 per month and $250 a month for the Enterprise. | Languages like Swift or Objective-C for iOS development, and Java or Kotlin for Android. IDEs such as Xcode for iOS and Android Studio, Flutter or React Native for cross-platform development  1.1  Mobile is once again pretty different here. Ontop of the universal web technologies we also take Swift, Objective-C for iOS and Kotlin for Android all into consideration. Cross platform development frameworks like React Native should also be utilized aswell as mobile emulation for testing. Due to different languages being used and different designs on iOS and Android it is recommended two separate teams be used, one for iOS and one for Android development. As previously mentioned Apple Developer runs about $100 a year and Google Play is $25 one time cost. |

## 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**: An appropriate operating platform that will allow The Gaming Room to expand Draw It or Lose It to other computing environments would be Windows, due to the platform's widespread use and compatibility with a wide range of hardware. Windows provides robust development tools and support for various programming languages, making it easier to port the game and maintain it across different platforms.

1.2

Windows Server has the support for horizontal and vertical scaling which makes it incredibly flexible and will help with handling increased computational loads when dealing with many players. It can also integrate seamlessly with Azure for cloud integration, as a hybrid approach would help best deal with the user base in an efficient manner while also remaining cost effective. Windows Server is also known for its reliability and great performance, which will be key when dealing with a web-based game that is highly accessible as the user base can get large fast. Advanced security features with windows like built-in firewall connections and security updates that occur on a regular basis help reduce the threat of malicious attacks. Windows Server is also compliant with many industry standards such as the General Data Protection Act and Payment Card Industry Security Standard. When it comes to handling sensitive data and collecting user data it must be done ethically and legally so Windows's compliance with these sorts of things makes it a great option. Windows is also a cost effective option as Microsoft has several flexible licensing options, allowing us to choose the most ideal one for Draw It or Lose It’s capabilities and needs. The widespread use of Windows Server also means that support is always available, if not from Microsoft directly there are many professionals who work with Windows Server full time so any major issues can be resolved in relatively short time. Windows Server is the most appropriate server platform that will allow The Gaming Room to expand Draw It or Lose It in the most efficient and cost effective manner possible.

1. **Operating Systems Architectures**: The Windows operating system architecture revolves around a layered model. At its core is the kernel managing system resources. Windows uses a modular approach and loads components dynamically. The system includes a Hardware Abstraction Layer for hardware compatibility as well which helps make it incredibly flexible for development. The broad hardware and software support for Windows makes it an enticing option.

1.2

Windows operates in two main modes: user mode and kernel mode. User mode is where applications run, while kernel mode is where the core operating system components and drivers operate. This separation ensures that applications cannot directly interfere with the core system, enhancing stability and security. In user mode, Windows includes various subsystems that provide support for different application environments, such as the Win32 subsystem for Windows applications, the POSIX subsystem for UNIX applications, and the OS/2 subsystem for OS/2 applications. This support allows Windows to run a wide variety of applications from different environments seamlessly. The Windows kernel is monolithic as well which means it includes a wide range of functionalities, such as memory management, process and thread management, and hardware abstraction. This design ensures high performance and efficient resource management. Within the kernel, Windows includes a set of executive services that provide core operating system functions, such as object management, process management, and security. These services are essential for the smooth operation of the system and the applications running on it. Another key factor to mention is the Hardware Abstraction Layer (HAL) in Windows. This provides a consistent interface for the kernel and device drivers. This abstraction allows Windows to run on a wide variety of hardware platforms without needing extensive modifications. By using the HAL, Windows can support different hardware configurations, including various types of processors, motherboards, and peripheral devices. This flexibility is essential for developing applications that can run on different hardware setups.

1. **Storage Management**: For Windows, an appropriate storage management system would be the Windows File System (NTFS). NTFS offers features like file compression, encryption, and access control. It also supports large file sizes and volumes, which is important for storing multimedia files and game data. NTFS is integrated into the Windows operating system which makes it seamless for compatibility and performance.

1.2

NTFS supports file compression, which allows the system to store files in a compressed format. This feature helps in reducing the amount of disk space used by game data, multimedia files, and other large files, optimizing the use of available storage. Files can be compressed automatically, and users can specify which files or folders to compress, offering flexibility in managing storage space. NTFS includes support for file and folder encryption through the Encrypting File System (EFS) as well. The encryption process is integrated seamlessly into the file system, making it easy to encrypt and decrypt files without requiring additional software or complicated procedures. NTFS uses a transaction-based logging mechanism, known as the Master File Table (MFT), to keep track of changes to the file system. This logging helps in recovering from crashes and maintaining data integrity by ensuring that file system operations are completed successfully. The file system includes self-healing features that can automatically detect and correct errors, reducing the risk of data corruption and minimizing the need for manual intervention. Overall the NTFS file system is an excellent choice for the storage management needs of The Gaming Room's Draw It or Lose It application.

1. **Memory Management**: Windows uses virtual memory to handle large amounts of data, memory protection mechanisms to ensure security and stability, and paging to optimize memory usage for the Draw It or Lose It software.

1.2

Memory management is key to optimizing performance for Draw It or Lose It. Windows uses virtual memory to extend the available physical memory (RAM) by utilizing disk space. This technique allows the system to handle larger datasets and more applications simultaneously by swapping data between RAM and the HDD or SSD. Each process in Windows is given a virtual address space, which isolates it from other processes. This isolation increases security and stability by preventing processes from interfering with each other's memory. Windows supports dynamic memory allocation as well which allows applications to request and release memory as needed. This flexibility is crucial for applications like Draw It or Lose It, which may have varying memory requirements based on user activity and game state. The operating system provides robust heap management features, including multiple heaps per process and efficient allocation/deallocation mechanisms. This ensures that memory is allocated and freed efficiently, reducing fragmentation and improving performance. Windows provides garbage collection for applications using C# like Draw It or Lose It. This automatic memory management system helps in reclaiming memory that is no longer in use, preventing memory leaks and optimizing memory utilization to the fullest. All of these memory management techniques ensure efficient memory utilization, security, and stability, supporting the game's performance and scalability as user demand grows.

1. **Distributed Systems and Networks**: Draw It or Lose It can communicate between various platforms through a distributed system architecture. Each platform would have its client app communicating with a central server managing game sessions, communication between clients, and storing game data. The system should implement redundancy and fault tolerance to handle network outages and connectivity issues, along with network protocols like TCP and UDP for reliable data transfer and real-time communication.

1.2

Since Draw it or Lose It is being developed for many different platforms, each platform would have a client application that communicates with a central server. This server manages game sessions, handles communication between clients, and stores game data. We also need to consider different network protocols like TCP. TCP ensures reliable data transfer between clients and the server, crucial for maintaining game state and synchronization. Another protocol UDP (user datagram protocol) can be used for real-time communication, such as sending quick updates and actions between players, where speed is prioritized over reliability. Another important consideration to make is to have the architecture support horizontal scaling which allows additional servers to be added as the number of players increases. Leveraging cloud services can also provide on-demand scaling and global reach, ensuring low latency and high performance for players worldwide. This approach ensures a seamless and reliable gaming experience, even in the face of connectivity challenges and growing user demand.

1. **Security**: Use secure authentication, encrypt sensitive data at all points, and implement access controls. Windows provides security features built into the operating system, like a built-in firewall, antivirus tools, and secure boot, which enhance the application's security.

1.2

Some security features to take into consideration are features like authentication services. Implementing multi-factor authentication ensures that users provide multiple forms of verification before gaining access. This reduces the risk of unauthorized access. The use of OAuth is another great secure form of verification. The Windows operating system includes a robust firewall that helps protect against unauthorized access and network attacks. Configuring the firewall properly helps to restrict incoming and outgoing traffic to only the needed services. Windows provides built-in antivirus and anti-malware tools, such as Windows Defender, that offer real-time protection against a wide range of threats. Another security implementation such as IDPS helps monitor and protect the network from malicious activities and potential breaches. Using a VPN for remote connections also helps ensure that data transmitted over public networks is encrypted and secure. The security capabilities of the Windows operating platform are vast. Authentication, encryption, and network security all provide a strong and secure environment for Draw It or Lose It.