

<Draw-It-or-Lose-It>

# **CS 230 Project Software Design Template**

Version 1.0

## 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 | <06/02/2024> | <Hanna-Parham> | <Filled out entirety of document.> |

**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 software design problem at hand is creating a web-based gaming application called “Draw It or Lose It”, which will be similar to the game “Win, Lose, or Draw”. The application will allow teams to compete by guessing phrases or titles based on rendered images. The solution will involve designing a system that supports multiple teams and players and allows for the uniqueness of game and team names, while also restricting the number of active game instances in the stored memory.

## Requirements

The requirements include:

* The game must be able to support multiple teams with multiple players
* The game and team names must be able to be unique
* Only one instance of a game can exist in the memory save at any given time

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

The design constraints would include:

* The game must be accessible over the internet which would require considerations for scalability, reliability and security measures to ensure privacy for users.
* Because each game name, team name and save file must be unique, there must be unique identifiers that are generated and managed efficiently.
* Because they are using rendered images, the images themselves must be able to render during gameplay, which would require fast and efficient processing and delivery.

## [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)

<Describe the UML class diagram provided below. Explain how the classes relate to each other. Identify any object-oriented programming principles that are demonstrated in the diagram and how they are used to fulfill the software requirements efficiently.>

The ProgramDriver class contains the main class, and it directly uses information from the SingletonTester class which restricts the instantiation of the game to one object. All of the classes directly inherit from the GameService class in order to effectively culminate information from the Game, Team, Player and Entity classes. Encapsulation is used in each class in order to protect internal data and maximize the visual of a clean interface.

**"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** | <Using Mac for hosting web applications is great for development because of its strong performance and security measures. Macs, however, are pricey and not as scalable as Linux servers, which are usually better for production due to their cost effectiveness and compatibility. For MacOS, there is typically a purchasing license required for each server instance which would have to be factored into deployment costs. As compared to Linux for example, Mac has higher upfront costs.> | <Linux is a great choice for hosting web based software applications due to its cost effectiveness, stability, security and scalability. Despite having a steep learning curve with many complexities, it has extensive community support and compatibility with a wide range of tools that allow it to be the preferred choice for production. Linux distributions are often open-source and free to use, so this cuts down costs significantly compared to other operating systems.> | <By choosing Windows for web hosting, it offers a very user friendly interface, broad compatibility and robust support. It provides regular updates and integrates well with Microsoft technologies which a lot of businesses rely on. It can however be costly to use, vulnerable to attacks, and has a hard time with licensing. Windows Server also requires purchasing licenses for each server instance like MacOs, so that would have to be factored into deployment costs as well. > | <While mobile devices are portable, accessible and easily personalized for web based applications, they also have screen size limitations, performance constraints, security concerns and heavily depend on network connectivity. The Apple App Store requires an annual membership fee of $99 USD, while the Google Play Store requires a one time registration fee of $25. Neither Android nor iOS offer server-based deployment methods, however backend services can be created and hosted on various server platforms such as Azure or Google Cloud to support mobile applications.> |
| **Client Side** | <Supporting multiple types of clients such as Mac users requires considering additional costs, time, and skills needed in developing and maintaining software for the macOs platform.> | <Supporting Linux clients needs a business to consider time, costs and skills needed to use the software. While costs may be lower due to Linux being open source, there would still need to be a fair bit of money allocated into development and extra time needed to test compatibility.> | <Like the other two platforms mentioned, there would of course need to be time, costs and skills needed for development to be considered, especially to ensure the cross compatibility across the different versions of the Windows OS. Platforms like React Native or Flutter should be used to make sure that the application is cross-platform compatible, and the code only has to be written once to save time and money.> | <Time, costs and expertise needed to handle mobile development would need to be considered, with development commonly finding extra expenses for mobile specific tools and distribution. It would also take more time to test across various platforms and devices as well. In order to ensure cross compatibility, the development process should rely on using cross-platform development frameworks like React Native or Flutter so the code only has to be written once to be deployed on multiple platforms.> |
| **Development Tools** | <Swift: Apple’s primary coding language would be used. As for IDEs and tools, Xcode, Terminal, CocoaPods, Homebrew and different text editors such as Visual Studio Code or Sublime Text could be used for editing purposes. While Xcode and Swift are free, there may be licensing costs applied to premium text editors or specific plugins; developer memberships may also be needed for advanced features or publishing apps.> | <JavaScript for front end development, and Python for back end IDEs and tools include: Visual Studio Code or Sublime Text, Node.js, Flask or Django, Git, Docker, MySQL and or Nginx/Apache. While core technologies are open-source, there may be costs for premium IDE features or plugins. Depending on the project complexity, there may be a need for a separate front-end and a back-end team.> | <All of what was mentioned for the Linux development tools would also work for Windows OS, with some cross platform differences to consider when launching the application on Windows versus Linux. While Visual Studio Code and Node.js are free, there may be licensing costs for specific plugins.> | <Swift for iOS, or Java/Kotlin for Android. The IDEs and tools would include Xcode for iOS, Android Studio for Android, Visual Studio Code, React Native and Flutter. There may need to be respective teams for each platform to optimize performance and user experience. Xcode and Android Studio are free, though there may be additional costs for enterprise features or support needed for potentially complex applications.> |

## 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**: <Recommend an appropriate operating platform that will allow The Gaming Room to expand Draw It or Lose It to other computing environments.>
   1. I would personally recommend using the Windows OS to expand to other computing environments because it is already compatible with Linux, iOS and Android which makes it a convenient choice for cross platform development, and, conveniently, it is also the OS I am most familiar with using and troubleshooting. By using Windows OS, the Gaming Room can ensure cross compatibility across numerous devices and operating systems while also simplifying the deployment process and reaching a broader audience.
2. **Operating Systems Architectures**: <Describe the details of the chosen operating platform architectures.>
   1. The Windows OS architecture is based off of the Windows NT kernel, featuring both user and kernel modes.The kernel is the core component of the OS responsible for managing system resources and providing essential services to user applications, while the user mode is a restricted operating mode where user applications and processes run. While user mode prioritizes application stability and security by preventing the direct manipulation and access of the more sensitive system components, the kernel acts as a bridge between software applications and the computer’s hardware to ensure an efficient and secure operation of the system.
3. **Storage Management**: <Identify an appropriate storage management system to be used with the recommended operating platform.>
   1. Because Windows was chosen, the storage management system used would be the Windows Storage Spaces feature which allows users to create storage pools by using physical disks or storage devices. Windows Storage Spaces also supports features like thin provisioning, storage tiering, and automatic disk repair which provides flexible and reliable storage management solutions.
4. **Memory Management**: <Explain how the recommended operating platform uses memory management techniques for the Draw It or Lose It software.>
   1. Windows uses a combination of physical RAM and disk space which ensures optimal performance in the event of physical memory being limited.
   2. Windows safeguards system memory for unauthorized use by using features like read only memory and access control lists.
   3. Windows allocates memory to running processes and applications based on demand which overall optimizes system performance.
   4. Windows compresses memory pages in RAM to conserve physical memory while also enhancing system responsiveness.
5. **Distributed Systems and Networks**: <Knowing that the client would like Draw It or Lose It to communicate between various platforms, explain how this may be accomplished with distributed software and the network that connects the devices. Consider the dependencies between the components within the distributed systems and networks (connectivity, outages, and so on).>
   1. To ensure communication between each platform for Draw It or Lose It, a client server architecture with a central server managing game logic would have to be implemented. Each platform would have to run a client application to interact with this server which would ensure reliable network connectivity, redundancy, and mechanisms to handle potential outages. Data synchronization would also have to be implemented in order to ensure consistent game data across multiple platforms.
6. **Security**: <Security is a must-have for the client. Explain how to protect user information on and between various platforms. Consider the user protection and security capabilities of the recommended operating platform.>
   1. Windows does offer built in security features such as Windows Defender, BitLocker and an advanced firewall to safeguard data. To protect users on and between multiple platforms, encryption would need to be a necessity for data transmission along with securing user authentication through the use of a multi-factor authenticator. Regular updates would also need to be given to the security measures to ensure nothing is outdated and or left vulnerable.