

<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 |
| --- | --- | --- | --- |
| 1.2 | 02/14/2023 | Kelly Perez | Adding in recommendations |
| 1.1 | 1/31/2023 | Kelly Perez | Adding additional information |
| 1.0 | 01/18/2023 | Kelly Perez | Information for initial design of Draw it or Lose it |

**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 expand their Android-only game, Draw It or Lose It, to multiple platforms. Teams compete to guess drawn images with time limits for both drawing and guessing. The game will have four rounds, each lasting one minute. This report will examine software development and distribution constraints. The solutions to these constraints ensure that the product expands the company's consumer footprint and market potential while remaining within their development budget.

## Requirements

*<* Please note: While this section is not being assessed, it will support your outline of the design constraints below. *In your summary, identify each of the client’s business and technical requirements in a clear and concise manner.>*

## [Design Constraints](#_2et92p0)

* Game must run on multiple platforms.
* Each game can have more than one team playing.
* Each team will have multiple players assigned to it.
* Game and team names must be unique to allow users to check whether a name is already in use.
* Only one instance of the game can exist in memory at any given time. This can be accomplished by creating unique identifiers for each instance of a game, team, or player.
* We must consider the best way to allocate memory efficiently as the game will be available on multiple platforms.
* The development and delivery will be executed within the budget constraints.

## [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)

UML diagrams show the game system's design. The entity class connects game, team, and player classes in the diagram. Arrows show that these classes inherit from a hierarchy. The graphic shows development classes, variables, and methods. The top-left programDriver Class connects to the singletonTester, suggesting code testing. To ensure only one game instance in memory. Complex methods power the game's functionality in the Gameservice class. The program requires unique game, team, and player classifications. Numbers show class relationships. Modifications to this diagram will guide development.

**"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** | Mac clients on a network can benefit from a MacOS server's program compatibility and graphical administrative interface. However, maintaining a Mac server can be expensive and may not be the best option for most enterprises that require considerable outside applications and customization, which could be costly in the end. | Easy on the wallet. The platform is difficult to maneuver about in. The shell makes it easier to set up the server and get access to it, but in order to utilize this platform with the level of proficiency essential for smooth maintenance, prior knowledge is required. | Comparatively higher software compatibility than other systems. Adaptable commands for setup, modification, and server terminal program access. Through the Windows Firewall and Security threat checks, Windows discovers security issues before they become severe problems. Windows is the most popular operating system among the general population. | It is generally impractical to access a server through a mobile device, yet it may be doable. Oracle is compatible with mobile servers. Most mobile devices run on the Android and iOS systems, thus familiarity with them will be required. |
| **Client Side** | Because the software is open source, the client-side price point can be rather inexpensive. The amount of time spent navigating the Mac server would depend on the user's familiarity with Apple software. A user with less experience may struggle with management, whereas a user with more expertise may succeed. | There is a need for considerable skill and time. Navigating Linux data is essential for system operation. Due to the requirement for understanding of the platform, the greatest expense for Linux users is time. | Being that Windows is one of the most widely used operating systems, the vast majority of the general public have the knowledge to navigate it. The cost would be relatively higher compared to other OS’s because of licensing for Microsoft. The platform is not open source, much like the Apple OS. | The mobile server option would be the most expensive since platforms are always evolving and need maintenance improvements. The vast majority of people use mobile devices; therefore the needed platform expertise will be innate, needing less time for the end user to learn how to navigate. |
| **Development Tools** | The languages hypertext markup language (HTML), cascading style sheets (CSS), C++, Python, and Java Script are all examples of front-end development tools. These are libraries that aid when developing frontends for websites. Additional integrated development environments (IDEs) that are helpful for software development include Eclipse and Visual Studios. | The languages hypertext markup language (HTML), cascading style sheets (CSS), C++, Java, Python, and Java Script are all examples of front-end development tools. JavaScript, Ruby, and PHP are other back-end programming technologies that may be utilized on a Linux platform. | The languages hypertext markup language (HTML), cascading style sheets (CSS), and Java Script are all examples of front-end development tools. Other development tools for Windows include the CMD and Eclipse. Languages include C++, Java, Python, and Ruby that assist with back-end development and coding. | The languages hypertext markup language (HTML), cascading style sheets (CSS), and Java Script are all examples of front-end development tools. Other development IDEs for mobile devices include PyCharm and Eclipse. Languages include C++, Java, Python, and PHP that assist with back-end development and coding. Knowledge of the Android and iOS platforms would be needed as most mobile devices use either of them. |

## 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**: A cloud-based platform like Amazon Web Services (AWS) or Microsoft Azure might be used to extend Draw It or Lose It. These platforms enable application deployment and scalability across desktop, online, and mobile platforms. The Gaming Room can grow the game to handle more users without worrying about hardware infrastructure using a cloud-based platform. Cloud platforms also provide security and flexibility in game development programming languages and frameworks.
2. **Operating Systems Architectures**: A cloud-based architecture would be a good choice for the recommended operating platform because Draw It or Lose It could be used in other computing environments. With a cloud-based architecture, the game could be deployed and run on more than one platform without needing a specific operating system. IaaS, PaaS, and SaaS are cloud-based architectures (SaaS). Each architecture has pros and cons. IaaS lets users construct and manage virtual computers and install applications in a virtualized infrastructure. Instead of managing infrastructure, PaaS offers a comprehensive platform for building, deploying, and maintaining applications. SaaS offers a comprehensive software solution that clients may access through a web browser without installing any software. Draw It or Lose It might expand with a PaaS architecture. This architecture would enable game creation, deployment, and administration without infrastructure management. This will let creators concentrate on game creation and assure flawless multiplatform play. For a game with many players, PaaS systems frequently provide capabilities for scaling and maintaining high availability.
3. **Storage Management**: The Gaming Room's storage management needs determine the possibilities. Amazon S3, a popular and scalable storage management technology, may meet their demands (Simple Storage Service). Amazon S3 stores any-sized items in the cloud. It provides a variety of storage classes for varying access needs and pricing models, as well as robust security and access controls to assure data confidentiality and integrity. Amazon S3 lets The Gaming Room store and manage player profiles, game assets, and user-generated material in a scalable and dependable way. Game clients on numerous platforms may access and update the data, and storage costs can be minimized depending on frequency and retention.
4. **Memory Management**: The cloud provider would manage Draw It or Lose Its memory using virtual memory and memory paging. Virtual memory lets the cloud provider extend physical memory by utilizing a piece of the hard disk. Memory paging lets the provider relocate seldom used memory pages to the hard disk, freeing up physical memory for active pages. The cloud provider would maximize application performance by buffering and caching frequently requested data in fast memory and limiting disk access. Draw It or Lose It would function easily and dependably on the cloud operating platform due to its efficient and scalable memory management.
5. **Distributed Systems and Networks**: Draw It or Lose It may connect across platforms using a distributed software architecture, which enables components to function together regardless of their location. APIs and web services may allow platform-to-platform connectivity and data transfer. A network infrastructure connecting devices and platforms would facilitate distributed software architecture. For rapid and responsive component communication, dependable and high-speed connection is needed. The system may employ redundant servers, load balancers, and other methods to stay highly available and robust to network failures and other difficulties. To guarantee that Draw It or Lose It can interact across many platforms and provide a smooth and responsive user experience, the distributed system and network architecture should be developed for scalability, dependability, and performance.
6. **Security**:

* **Encryption**: All user data, including login credentials, and personal information, should be encrypted using secure encryption algorithms. This will help protect the data in case of any data breaches or unauthorized access.
* **Access control:** Access to user data should be restricted to authorized personnel only. This can be achieved by implementing role-based access control, where users are assigned specific roles and privileges based on their level of authority.
* **Secure communication protocols**: The communication between different platforms should be secured using secure communication protocols, such as SSL/TLS. This will help protect user data from interception and eavesdropping.
* **Firewalls and intrusion detection systems**: Firewalls and intrusion detection systems can be used to prevent unauthorized access to the network and detect any attempts to breach the security of the system.
* **Regular security audits**: Regular security audits should be conducted to identify any security vulnerabilities and ensure that the security measures are up-to-date and effective.