

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.3

## 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 | 02/05/24 | Eric Thomas | Initial Version |
| 1.1 | 02/16/24 | Eric Thomas | Evaluation |
| 1.2 | 2/26/24 | Eric Thomas | Recommendations |

**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, a new client, seeks to expand the reach of their game "Draw It or Lose It" beyond its current Android platform. They aim to develop a web-based version accessible across multiple operating systems, including MacOS, Linux, and Windows. This multiplayer game revolves around teams guessing drawn images within specified time limits across four one-minute rounds of play.

## 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)

Cross-platform Compatibility: The game, originally an Android app, must transition to a web application accessible across various operating systems. Utilizing a REST API for communication via HTTP will ensure compatibility without language restrictions.

User Interface Consistency: Retaining or enhancing the current Android interface design is essential to maintain user familiarity. Options include replicating the existing interface or upgrading it for a refreshed gaming experience.

Multi-team and Multi-player Support: The game must accommodate multiple teams with multiple players each, necessitating a robust client-server architecture capable of handling concurrent players efficiently.

Unique Identifiers and Memory Management: Ensuring uniqueness of game and team names while managing memory allocation is crucial. Incorporating unique identifiers for games, teams, and players, along with platform-specific player IDs, is necessary for seamless cross-platform functionality.

Image Usage and Copyright Compliance: The game will incorporate images for gameplay, requiring adaptation of existing Android images for compatibility across platforms. Obtaining licenses or copyright permissions for additional images is imperative during the design phase to ensure legal compliance.

## [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)

The Unified Modeling Language (UML) diagram serves as a visual representation of the game system's design. It outlines the relationships between entity classes such as game, team, and player, indicating inheritance from a superclass. The diagram illustrates the classes, variables, and methods utilized in development. The programDriver class, located in the top left corner, references the singletonTester, demonstrating its utilization for testing code restrictions, ensuring only one instance of the game exists in memory. The GameService class houses complex methods essential for the game's functionality, including the requirement for unique games, teams, and players, reflected in the diagram's design. Interconnecting lines signify associations between classes, with numbers denoting the quantity of associations within each class. For instance, the GameService class may have zero or more associated games, teams, and players. This UML diagram guides the development process, allowing for adjustments to be made as needed, ultimately leading to the creation of the final product.

**"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** | provides server-based environments that present significant benefits for Mac clients within a network. These advantages include comprehensive support for Mac applications and straightforward general administration facilitated by an intuitive graphic interface. However, the drawbacks of employing a Mac server include potential high maintenance costs and limitations for larger enterprises heavily dependent on third-party programs and customizations. | offers numerous advantages for web hosting, with even major companies like Google utilizing Linux servers. One key advantage is its cost-effectiveness as a free and open-source platform, reducing deployment and tool usage expenses. Additionally, Linux's high level of customizability enhances security, allowing organizations to tailor security measures to their needs. Furthermore, Linux web hosting supports popular programming languages like Python, PHP, Perl, and Ruby. | Windows is a comprehensive operating system, albeit proprietary and requiring licensing fees. Despite potential expenses, hosting web servers on Windows offers advantages such as extensive support for applications and third-party software, straightforward patch and hardware updates, and full compatibility with development tools like ASP.NET and databases like MySQL for developers familiar with the platform. | While utilizing mobile devices as web servers is uncommon and potentially impractical, companies like Oracle offer solutions for mobile server-side implementation. Oracle's Database Mobile Server facilitates the management of applications, users, devices, and data across extensive deployments of mobile or remote devices. The advantages of this Oracle database structure include support for iOS and Android development tools. |
| **Client Side** | Costly for users. Requires moderate time and expertise. Accurate skills necessary to navigate the operating system. | Considerable expertise and time investment is necessary to utilize Linux. Proficiency with Linux data is essential for operating the system. Linux users may face significant costs. | More costly compared to Linux systems. Simple to grasp and support a Windows setup. Minimal expertise required. | Offer flexibility for clients and developers to access updates from anywhere. Slightly more challenging to implement compared to other devices. |
| **Development Tools** | JavaScript, HTML, CSS. Frontend development. Pycharm or VS Code | JavaScript, HTML, CSS. Frontend development | JavaScript, HTML, CSS. Frontend development | JavaScript, HTML, CSS. Frontend development |

## 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**: The optimal operating platform for expanding Draw It or Lose It to various computing environments is Windows. Windows is preferred as it offers a wide array of IDEs for development work, ensuring ample options for the project.
2. **Operating Systems Architectures**: Microsoft Windows, developed and released by Microsoft, is a graphical operating system offering capabilities to store files, execute software, engage in gaming, watch videos, and access the internet.
3. **Storage Management**: Windows 10 includes a useful feature known as Storage Sense, enabling users to inspect and oversee files stored on their hard drive, as well as monitor the amount of space they occupy.
4. **Memory Management**: Windows 10's Storage Sense feature enables the storage and organization of Draw It or Lose It photos and player data, ensuring they are securely kept together in a unified memory space.
5. **Distributed Systems and Networks**: Network-based multi-user interaction systems, such as network games, often involve a shared database accessed by physically distributed players who interact over the network. Currently, developers of network games must create the shared database and inter-player communication systems from the ground up.
6. **Security**: Windows includes built-in security protection software. However, for enhanced security of user data and information, it is advisable to utilize additional security measures from external sources.