

Draw It or Lose It

# **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 | 05/25/24 | Jacob Ohara | Completed Executive Summary, Design Constraints, and Domain Model |

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 client, The Gaming Room wants to develop a web-based game that serves multiple platforms based on their current game, Draw It or Lose It, which is currently available in an Android app only. The client has specific software requirements which include: to have one or more teams involved, each team will have multiple players, game and team names must be unique, allow users to check whether a name is in use when choosing a team name, only one instance of the game can exist in memory at any given time.

## 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)

1. Complexity of Cross-Platform Compatibility:

The game must be accessible and functional across different operating platforms, devices, and web browsers.

2. Restricted UI Design:

The game UI must be user friendly and consistent across all platforms, devices, and web browsers.

3. Unique Names:

System to check game and team names are unique, enables users the ability to retrieve relative information when choosing their team names.

4. Singleton Design:

Only one instance can exist in memory at a time and allow a single point of access.

5. Scalability:

The game must be capable of handling large amounts of players without compromised performance.

## [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 Gaming Room’s domain model has the following entities and relationships:

• Program Driver: contains the main method

• Singleton Tester

• Entity: This is a superclass

• Game Service: This class is a singleton

• Game: This class contains a list of teams

• Team: This class represents a list of players for a team

• Player: This class represents a player.

The OOP in this program includes encapsulation with the methods in the class, inheritance from the entity class, and polymorphism through the overrides.

"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 must 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 | MacOS offers many deployment methods for hosting the website including: macOS Server, VPS (virtual private server) hosting, EC2 through AWS. All options are compatible with different web browsers and devices and are designed to interpret content consistently through HTML, CSS, and JavaScript. Cost can range depending on resources needed and region, additionally they aren’t commonly used in this space making finding experienced team members less cost effective. | Linux is one of the most popular OS for web hosting and server-based deployment. LAMP Stack is a popular platform that uses Linux, Apache, MySQL, and PHP. This resource is open source meaning it is free for use. Additionally, Apache HTTP is the most currently used web server software which is also free to use. Since Linux is very widely used, finding experienced team members will be more cost effective. | Windows offers deployment methods from IIS, and Azure. Both options can be used to reach every target audience. IIS is included with windows. Azure allows App service to host a web app, API app and mobile app back end. Additionally, they have platforms for live games and real time analytics tools. Azure must pay for what you need model making it cost effective additionally windows is the most widely used OS for gaming so there are experienced developers already in the space. | It is not common to host webservers on a mobile device due to lack of security and performance. For IOS it does not offer any options to host web services, while android has apps that can be used to allow you to install packages to host web servers. However, they are mainly used for testing and not functionality. |
| Client Side | High cost due to specialized expertise in development process utilizing Xcode and Swift. Additionally, leading to longer development times to support testing for multiple types of clients. Apple has specific design guidelines that must be followed throughout the testing across different browsers and devices. | Highly Secure, Stable, and flexible. Currently extremely popular, however expertise in HTML, CSS, JavaScript are necessary to test across different devices and browsers. Widely used and constantly updated making it easier to find developers with the necessary expertise. | Long cycles of development and testing across different clients. Tools like .NET, C sharp and Visual Studio are most used for client-side development. Like apple windows has design guidelines developers must follow through testing making it more expensive than Linux. | High cost because of the specialized experience dealing with IOS and Android. Developers need to be experienced in Java mainly for android and Swift for IOS as they are the main tools being used in development for those Apps. |
| Development Tools | The most common programming languages for mac are Swift, and JavaScript. Swift is used for iOS and mac apps, while JavaScript and HTML are used for web development. Mac’s official IDE is Xcode which is free and comes included however Visual Studio Code is also popular and free as well. | The most common programming languages for Linux are Java, JavaScript, C++. With Java and JavaScript having the most versatility. Popular IDEs in Linux are Visual Studio Code and Sublime Text; however Sublime Text does have licensing fees. | JavaScript, Java, and C++ are common in the development for windows as well as HTML for web development. Popular windows IDEs include Visual Studio Code, which is free, and Microsoft Visual Studio that requires a licensing fee. | Mobile devices vary Swift is mainly used for iOS and Java is mainly used for Android development. JavaScript, HTML and CSS can be used for cross platform mobile apps. The most popular IDEs include Xcode for iOS and Android Studio and IntelliJ for android. Xcode is free, and IntelliJ is around $20 a month. |

## 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**: For the expansion of Draw It or Lose It I would suggest Linux. Linux is well known for its performance, stability and strong community support. Linux is open source making it free to use and reducing overall cost additionally most cloud supporters offer Linux support.
2. **Operating Systems Architectures**: Linux utilizes a Kernel based architecture. Provides a single large binary that includes drivers and services. Loadable kernel modules that allow for addition of features without the need for system reboots. User Space Application like Daemon which handles web servers and databases as well as other services, and containerization with Docker.
3. **Storage Management:** An appropriate storage management system for the application will include the use of partitioning, filesystems, mount points, and GUID Partition Table. Partitioning will divide a disk into different useable and manageable sections, the filesystems will organize and manage the data stored in each individual partition, the mount points will be used as directories to point to where partitions and filesystems are attached and allow access to data. GUID Partition Table (GPT) is used as a layout of the partition tables of the physical storage components.
4. **Memory Management:** For Draw it or lose it Linux can use virtual memory and paging, demand paging and swapping, and kernel same page merging. Virtual memory will ensure that the game runs in a virtual address. Paging will take the virtual address and translate them to physical addresses. Then keep track of the mapping between the virtual and physical memory. Demand paging will allow pages to be loaded in physical memory only when needed, reducing the overall amount of memory used. Page swapping will allow for when there is limited space inactive pages in use will be swapped out into free space freeing up RAM. Same Page merging will save memory usage by merging identical pages into a single shared page. Using these techniques will provide a responsive and consistent user experience with the application.
5. **Distributed Systems and Networks:** To ensure communication between platforms, we will need client applications able to run on multiple platforms and backend services hosted on cloud servers. Database servers to hold player data, game instances, and history; and include the use REST APIs. For the Network and Communication, we will need to utilize HTTP/HTTPS for client communication, WebSocket and geographically placed servers. For contingency handling of connectivity and outage issues, there will need to be database replication, primary and backup servers with a failover to automatically switch when one fails, and memory caching to reduce the load on the database.
6. **Security:** To ensure the application's security, there is a need to encrypt data stored in the database and encrypt the data sent between the clients and servers. The use of RBAC to define permissions for accessing resources. Linux specifically uses Pluggable Access Modules for user authentication and authorization. Data encryption with TLS/SSL simplified firewall management that is frequently updated with security patches. Additionally, the use of Kernel management and paging promotes isolation keeping data and application permissions isolated.