

Draw It or Lose It

# **CS 230 Project Software Design Document**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template**](#_l6ti7uoag22u)1

[**Table of Contents**](#_30j0zll)2

[**Document Revision History**](#_grjogdjh5fi8)2

[**Executive Summary**](#_sbfa50wo7nsh)3

[**Design Constraints**](#_2et92p0)3

[**System Architecture View**](#_ilbxbyevv6b6)3

[**Domain Model**](#_8h2ehzxfam4o)3

[**Evaluation**](#_2o15spng8stw)3

[**Recommendations**](#_m8aleynsvzvc)5

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 06/17/22 | Mike Berry | First version; Initial design |

**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 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.

## [Design Constraints](#_2et92p0)

* The game will have the ability to have one or more teams involved.
* 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 in use when choosing a team name.
* 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.

## [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)

In the UML diagram below, the abstract class Entity (abstraction; since one does not create an instance of Entity, but of its child classes instead. This helps reduce redundancy and increases reusability) has private instance variables “id”, of type long, and “name”, of type String (encapsulation of variables using the keyword “private” limits access to them). It also has a private default constructor and a public overloaded constructor accepting “id” and “name” as arguments for its two parameters. As for methods, it has “getId()”, “getName()” (getters and setters are examples of encapsulation), and “toString()”, of type long, String, and String respectively.

There are three classes that inherit from the Entity class; Game, Team, and Player (inheritance; makes code less redundant and reusable). The Game class can have zero to many Team instances and the Team class can have zero to many Player instances (these are class associations). Within the Game class there is a private instance variable (encapsulation) “teams” of type List. There is also a public constructor and methods “addTeam()” and “toString()”, of types Team and String respectively. Class Team has a private instance variable “players” of type List (encapsulation), a public constructor, taking “id” and “name” as arguments, and public methods “addPlayer()” and “toString()”, of types Player and String respectively. The Player class has a public constructor, taking “id” and “name” as arguments, which are of type “long” and String respectively, as well as a public method “toString()” of type String.

There is a class called “GameService”, which is associated with the Game class, and can have zero to many instances of the Game class. It contains private instance variables “games”, “nextGameId”, “nextPlayerId”, “nextTeamId”, and “service”, of types List<Game>, long, long, long, and GameService respectively (encapsulation). Also, it has a private constructor and public methods “getInstance()”, “addGame()”, two variations of “getGame()” with different parameters (polymorphism; makes the code more dynamic, clean, and reusable), “getGameCount()”, “getNextPlayerId()”, and “getNextTeamId()”, of types GameService, Game, Game, Game, int, long, and long respectively (encapsulation). Finally, there is the ProgramDriver class that uses the SingletonTester class (directed association). The ProgramDriver class contains the public method “main()” and the SingletonTester class contains the public method “testSingleton()”.

**"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** | Offers a server-based deployment method. Server configuration commands are versatile and easy to learn. There is a cost incurred for software licensing. | Offers a server-based deployment method. Open source. Similar versatility to macOS but distributions of Linux are free. | Offers a server-based deployment method. Microsoft offers a more larger range of software and is more user-friendly. There is a cost incurred for software licensing. | No server-based deployment method. Needs a backend to help with tracking user and app data, event scheduling, and applying changes to the application. Will require user of one of the other three operating platforms and whatever costs come with them. |
| **Client Side** | There is a cost incurred approximately the same as Microsoft Windows. Takes some intermediate computer knowledge to become fluent in operations. | Can be daunting to learn. Can use a GUI but usually operated through the command line which can be daunting for beginners. Takes some time to become comfortable. | Very user-friendly and cost is similar to macOS. GUI interface is familiar to most PC users. | More difficult to develop on than the other operating platforms, but offers more freedom for the end user due to mobility. |
| **Development Tools** | All of these operating platforms can utilize the more common languages for development like HTML, CSS, JavaScript, Python,  Java, Ruby, and php.  Development tools used include Eclipse, PyCharm, Visual studio.  Development with Java is recommended to ensure the application is compatible with all web browser platforms and mobile devices. | All of these operating platforms can utilize the more common languages for development like HTML, CSS, JavaScript, Python,  Java, Ruby, and php.  Development tools used include Eclipse, PyCharm, Visual studio.  Development with Java is recommended to ensure the application is compatible with all web browser platforms and mobile devices. | All of these operating platforms can utilize the more common languages for development like HTML, CSS, JavaScript, Python,  Java, Ruby, and php.  Development tools used include Eclipse, PyCharm, Visual studio.  Development with Java is recommended to ensure the application is compatible with all web browser platforms and mobile devices. | All of these operating platforms can utilize the more common languages for development like HTML, CSS, JavaScript, Python,  Java, Ruby, and php.  Development tools used include Eclipse, PyCharm, Visual studio.  Development with Java is recommended to ensure the application is compatible with all web browser platforms and mobile devices. |

## 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 recommendation for an operating platform that will allow expansion to other computing environments is to utilize Windows Server 2022 Datacenter Azure Edition.

1. **Operating Systems Architectures:**

Windows Server 2022 Datacenter Azure Edition is user-friendly, offers a diverse set of software packages, and has improved security and networking to meet the demands of the 21st Century as well as Azure hybrid capabilities. Azure will help cut costs and improve scalability. Windows has command line capability but leans heavier into using the more intuitive GUI (Graphical User Interface), allowing for advanced manipulation of settings and configurations without the need to learn a new complex system.

1. **Storage Management**:

Advanced storage upgrades have been implemented in the 2022 version of Windows Server. These features include advanced caching, improved hard drive repairs in the event of server restart or hardware failure, and a feature called ReFS Snapshots, which takes real time snapshots of data regardless of its size for faster repair and resychronization.

1. **Memory Management**:

Windows Server 2022 supports up to 48TB of memory and 2,048 cores on 64 sockets for improved scalability. Also, with the release of Windows Server 2022, the cache manager is now NUMA aware. NUMA, or non-uniform memory access, platforms are server platforms using more than one system bus. These platforms can utilize multiple processors on a single motherboard, and all processors can access all the memory on the board, optimizing cache IO workloads.

1. Distributed Systems and Networks

Windows Server 2022 has upgraded networking and communication capabilities over its previous edition. Among those improvements are UDP and TCP performance, and Hyper-V virtual switch improvements. Virtual networking is fully supported. One of the advances in UDP is the implementation of USO (UDP Segmentation Offloading) which lessens the load on the CPU. UDP is becoming more popular in online streaming and gaming and Windows Server 2022 brings UDP reliability and performance up to the level of TCP. Windows Server also features RACK and Hystart++, which provide smoother network data flow, with better performance at higher speeds.

1. Security:

Windows utilizes standard security features such as password-protected user accounts, role-based system permissions, VPN capabilities, and advanced encryption procedures. In addition, Windows Server 2022 has made improvements to transport with HTTPS and TLS 1.3 enabled by default. Connections are more secure due to the implementation of encrypted DNS name resolution requests with DNS-over-HTTPS. Windows Server 2022 also uses updated Server Message Block (SMB) AES-256 encryption for the most security-conscious, East-West SMB encryption controls for internal cluster communications, SMB Direct and RDMA encryption, and SMB over QUIC.

References

Bisson, S. (2022, June 10). *Windows Server 2022: A cheat sheet*. TechRepublic. https://www.techrepublic.com/article/windows-server-2022-a-cheat-sheet/

D. (2022, January 27). *What’s new in Windows Server 2022*. Microsoft Docs. https://docs.microsoft.com/en-us/windows-server/get-started/whats-new-in-windows-server-2022

Shethi, S. (2022, April 19). *10 New Things in Windows Server 2022 to Know*. Geekflare. <https://geekflare.com/new-features-in-windows-server-2022/>