

# 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 | 03/19/23 | Sam Blanton | Added a parent class entity and added teams and players. |

**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 has requested a web-based version of their game, Draw It or Lose It, where teams compete to guess what is being drawn. Our solution includes a software design that meets the client's requirements of having one or more teams involved with multiple players assigned to each team. The game and team names must be unique, and only one instance of the game can exist in memory at any given time.

To accomplish this, we propose creating unique identifiers for each instance of a game, team, or player. We will use Java to develop the application, utilizing the Singleton pattern to ensure that only one instance of the game exists in memory. This allows us to organize games and have multiple teams and also multiple players for each team. This gives us the structure to add their game functionality on top of it.

Our proposed solution meets the client's software requirements and will streamline the development process. By following this software design, we will create a robust and efficient web-based game application that meets the client's needs.

## 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)

Web-based environment: The game application must be designed for the web-based distributed environment, which means the application must be accessible through a web browser and must support multiple platforms.

Multiple teams: The game must support multiple teams, and each team can have multiple players.

Unique game, team, and player names: Game and team names must be unique to allow users to check whether a name is in use when choosing a team name. Similarly, player names must be unique within a team.

Single instance: Only one instance of the game can exist in memory at any given time. This means that each instance of a game, team, or player must have a unique identifier.

Implications:

Web-based environment: The front end of the game must be made with HTML, CSS, JavaScript, and have our java code on a server to give the game functionality. The game must also be designed to be responsive and adaptable to various screen sizes and platforms.

Multiple teams: The application must be designed to support multiple teams and provide an interface for team creation, joining, and management. The application must also be able to handle team-based game logic and scoring.

Unique names: The application must have a mechanism to check whether a game, team, or player name is already in use and prevent duplicate names. This requires implementing a database system to store and retrieve game, team, and player data.

Single instance: The application must ensure that only one instance of the game is running at any given time. This can be achieved by using a singleton class.

## [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)

This UML class shows inheritance by having the Entity parent class. This is because Game, Team, and Player all inherit from Entity. The GameService class is the backbone of the program because it is what keeps track of games and it does this well because it is a singleton class so it is easy to track instances. Each class has several public methods that all tie together to make the program work.

**"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 is not a good option for web hosting because Mac does not sell server grade equipment anymore. This along with an operating system aimed at consumers is a bad combo. There are ways to deploy a website but at the moment Apple has moved to an SOC arm based computer making upgrading impossible and you are limited to consumer chips. Licenses are free with each mac you purchase, but MacOS is not a server OS even if it is capable of performing server tasks | Linux is an ideal option for web hosting because the OS is free to use commercially, but it can be harder to set up and maintain. There are 100s of different ways to deploy a web server with linux meaning there is lots of documentation and options so that you can find what suits your needs. The OS also takes few resources to run meaning more bandwidth for the server applications | Windows server is a fine option for hosting. The only real draw back is Windows Server licenses cost thousands of dollars making it hard to recommend over linux. However, if you are looking to build your server application using a GUI this is your best option. It is easier to set up and maintain than linux, but that comes at a licensing cost. Not only do you have to purchase each individual server license you must also pay for how many accounts you would like to have, it is very pay to play. You do get the full power of Microsoft behind you meaning you get many security updates that are easy to install | No. between hardware limitations and OS limitations this is a no across the board for web hosting. You can not use IOS at all it provides no server features at all. If you had an itching for pain you can use android because it is an operating system that is built on the linux kernel and could be configured to host a website, however you will run into many issues because most android phones use ARM architectures and that has many compatibility issues. You would also have to set up a complex network of load balancing because you would need a lot of phones. But the OS is completely free to use because it is based on linux |
| **Client Side** | Because it is browser based this is no hinderance as long as a modern browser is used. Because the web interface will be based on Android app, a responsive HTML page will be required with CSS that can accommodate for any size screen and still look professional and be intuitive to use. Development should also make sure not to use any CSS attributes that are not supported on all popular browsers as this would break the UI if a client uses a browser that doesn’t support the latest CSS features. It is also important to take note of Safari that can have some weird quirks when it comes to CSS it is important to make sure it is well tested on safari as well as chrome. | Because it is web based the reasoning for the three desktop OS’ are the same because the limiting factor is just screen size and the browser being used. | Because it is web based the reasoning for the three desktop OS’ are the same because the limiting factor is just screen size and the browser being used. | For Mobile it is important that the HTML and CSS are capable of handling the smaller screen sizes in an intuitive way. It is also important that you account for Safari on IOS because you have to use some webkit CSS tags that are not supported in most browsers but is required to accomplish some things in Safari. Because IOS is such a large percent of the market it is imperative that the website works great of Safari and Chrome, The most popular browser for Android phones. |
| **Development Tools** | Mac is well suited for development. VS Code with the Java, python, CSS, HTML, Javascript plug ins makes this a powerful way to develop code. The bash terminal is also a plus. This is also the only way to develop for safari in particular that has its own quirks. You are also able to use Xcode to how it would look on an iphone thanks to its ability to emulate any iphone. To build out a fleshed out server that can host a web based game will likely need mulitiple development teams. At the very least you will need a team for backend development and a team for front end development because these are two very separate programming skills that must work together. Though there are development tools that cost money like jet brains IDEs it is very possible to use only free development tools. | Linux is well suited for development. VS Code with the Java plug in is a powerful way to develop code. The bash terminal is also a plus. You will run into more technical difficulties using a linux distro IMO. It can very difficult to accomplish even simple tasks on linux. To build out a fleshed out server that can host a web based game will likely need mulitiple development teams. At the very least you will need a team for backend development and a team for front end development because these are two very separate programming skills that must work together. . Though there are development tools that cost money like jet brains IDEs it is very possible to use only free development tools. Although Xcode is not available on linux | Windows is a great option for development because VS Code is also on this platform. I personally miss the bash terminal when I use windows but that is subjective. To build out a fleshed out server that can host a web based game will likely need mulitiple development teams. At the very least you will need a team for backend development and a team for front end development because these are two very separate programming skills that must work together. . Though there are development tools that cost money like jet brains IDEs it is very possible to use only free development tools. Although Xcode is not available on Windows. Also There is a license fee for windows 11 that will be required to develop on a windows plaform | No. There are no real IDE’s for mobile you have to resort to a cloud solution which is not optimal when paired with the tiny screen. |

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**

I recommend Linux for the server side operating system for The Gaming Room. Linux is an open-source, flexible, and highly customizable operating system that provides excellent support for network and server environments. Linux is widely used in the gaming industry and has a broad range of tools and libraries that are compatible with various gaming software. It is the most common server OS and there is a good reason for it, it is easy to scale and has the ability to do any task you could need.

1. **Operating Systems Architectures**:

I recommend x86 to be the architecture of your Linux server. There are simply more numerous and stable drivers with x86, and because you shouldn’t be hindered by power then x86 is the better performer per dollar. It is the most common Architecture for server equipment and is easier to source the equipment. There are many libraries and tools that have simply not been developed on ARM yet.

1. **Storage Management**:

The recommended storage management system for Linux is the Ext4 file system. The Ext4 file system is a mature, stable, and reliable file system that supports large file sizes and provides excellent performance. It also supports journaling, which helps to prevent data loss in the event of a system crash or power failure. This is very important for game servers because this will keep data safe even in the event of power loss.

1. **Memory Management**:

Linux has several tools for memory management, including demand paging and swapping. Demand paging enables the operating system to load only the required parts of an application into memory, reducing the amount of memory needed. Swapping allows the operating system to move inactive processes out of memory to disk, freeing up memory for active processes. These two features can lower the amount of RAM you need to buy or just make sure your performance doesn’t drop too much if you do run out of RAM.

1. **Distributed Systems and Networks**:

To enable communication between various platforms, The Gaming Room can use a distributed software architecture that allows for decentralized communication between different devices. The architecture should be designed in such a way that it can handle connectivity issues and outages, ensuring the uninterrupted flow of data. The use of RESTful APIs can be used to enable communication between different platforms.

1. **Security**:

To protect user information on and between various platforms, The Gaming Room should implement several security measures, including data encryption, authentication, and access control. Linux provides several security features, including firewall, SELinux, and AppArmor, which can be used to secure the system. The use of SSL/TLS protocols for communication can be used to secure data in transit. The implementation of multi-factor authentication can be used to enhance user protection.