

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.1

## Table of Contents

[**CS 230 Project Software Design Template**](#_heading=h.gjdgxs)1

[**Table of Contents**](#_heading=h.30j0zll)2

[**Document Revision History**](#_heading=h.3znysh7)2

[**Executive Summary**](#_heading=h.2et92p0)3

[**Design Constraints**](#_heading=h.tyjcwt)3

[**System Architecture View**](#_heading=h.3dy6vkm)3

[**Domain Model**](#_heading=h.1t3h5sf)3

[**Evaluation**](#_heading=h.2s8eyo1)3

[**Recommendations**](#_heading=h.3rdcrjn)5

## [Document Revision History](#_heading=h.3znysh7)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 01/16/22 | Taylor Roark | Initial version and requirements |
| 1.1 | 02/12/22 | Taylor Roark | Added development requirements and system 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](#_heading=h.2et92p0)

The Gaming Room seeks to develop a web-based game based on their current Android-based game, Draw It or Lose It. They have requested help in setting up the environment for the game. They have requested it be available across multiple platforms as well.

## [Design Constraints](#_heading=h.tyjcwt)

The game must be web-based and serve multiple platforms. The game must have the option for one or more teams, and each team can have multiple players assigned. There must be a check in place that ensures that game or team names to be unique, and allows users to see as such. Finally, there must only be one instance of a game existing in memory at a time, and will be accomplished by creating unique identifiers for each instance of a game, team, or player.

## [System Architecture View](#_heading=h.3dy6vkm)

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](#_heading=h.1t3h5sf)

The class ProgramDriver has a direct association and uses the class SingletonTester. The classes Game, Team, and Player all inherit from the class Entity, which is an example of the object-oriented programming principle Inheritance. This helps minimize duplicate code within the program and increases efficiency. Within a GameService instance, there might be none or many instances of Game. With there being an instance of the class Game, there might be none or many instances of the class Team. Finally, within an instance of the class Team, there might be none or many instances of the class Player.

**"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](#_heading=h.2s8eyo1)

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 LDAP, DAP and Cloud support.  -Based on Unix, making development easier with a comfortable terminal and convenient commands. | -Offers LDAP and Cloud support.  -Most popular OS for running web servers.  -Uses open source software which gives multiple hosting options.  -Based on Unix, making development easier with a comfortable terminal and convenient commands.  -Not all devices run well with Linux.  - | -Offers LDAP, DAP and Cloud support.  -Difficult to use for advanced development due to PowerShell.  -Websites with a high volume of activity can have problems with Windows hosting.  -Windows servers require frequent rebooting to run properly. | -Has LDAP support.  -Faster than web apps  -Offers offline support. |
| **Client Side** | -macOS machines tend to be vastly more expensive than other OS machines.  -Only OS to support Safari.  -Moderate expertise/experience and time needed. | -Requires the most expertise and time out of the OS.  -Least expensive out of the OS(it is open-source & free).  -Linux hosting is not compatible with Windows applications which can be a major issue if your company requires the use of windows based applications on a server. | -Windows machines tend to be cheaper than other OS machines.  -Minimum expertise and time needed, very easy to operate. | -Moderate expertise needed.  -May take longer due to app approval processes.  -May cost more than web apps due to differing OS(IOS, Android) on different app stores. |
| **Development Tools** | -Vast amount of software and tools available for graphics rendering.  -HTML/CSS/JavaScript and supporting libraries to support frontend development.  -General purpose programming languages such as Python, Java, PHP, Ruby on Rails.Rails.  -Licensing for Mac and other specific tools may be needed. | -Open source with tons of tools available.  -HTML/CSS/JavaScript and supporting libraries to support frontend development.  -General purpose programming languages such as Python, Java, PHP, Ruby on Rails.ls. | -Vast amount of software and tools available for graphics rendering, much more so than Mac.  -HTML/CSS/JavaScript and supporting libraries to support frontend development.  -Licensing for Windows specific and other tools may be needed.  -General purpose programming languages such as Python, Java, PHP, Ruby on Rails. | -Not many tools available.  -Bootstrap(CSS), DevTools(HTML, JavaScript, CSS), Blisk. |

## 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 Linux operating platform will best allow The Gaming Room to expand Draw It or Lose It to other computing environments, when compared to other operating systems.
2. **Operating Systems Architectures**: The Linux system works on four different layers. Firstly, the Hardware layer, which consists of all physical devices attached to the system. Next, there is the Kernel, which is the core of all Linux operating systems, and is what directly interacts with the hardware. Thirdly is the Shell, which takes input from users and sends it to the Kernel, and receives output from the Kernel. Finally, there is the applications, which are the utility programs that run on the Shell.
3. **Storage Management**:In combination with using Linux operating systems, I recommend using XFS for storage management. XFS specializes in large data files, which is essential when looking at the large library of images used for the Draw It or Lose It software.
4. **Memory Management**: The core of a Linux computer is CPU and RAM. Process information will be copied from RAM to CPU and the CPU will build its cache. Cache is extremely important in Linux for various reasons regarding memory management. A user will be served RAM when a user is requesting information from the hard disk. During the process of the information being copied from the hard disk, the information is placed in “page cache”. Information is stored to make the process rapidly faster if the same information is needed to be called on again. Virtual memory is also essential in memory management of Linux systems. It is used by the Kernel(the third shell of the Linux operating system architecture) to let programs make a memory reservation. When using this specific memory, none of the other programs can use this memory.
5. **Distributed Systems and Networks**: Using a client-server model, which is what is to be used for the Draw It or Lose It software, the client requests a service from the server. The client will establish a connection via the internet, and the server will connect to the client to fulfill the request. The connection will then terminate after the client’s request has been fulfilled.
6. **Security**: With distributing to various platforms, it is essential that in the base code there is user authentication. This can be done with basic user authentication via username/passcode confirmation and a secure class with iterators and singleton patterns. Within the Linux operating system itself, there are various key security features. This includes a user-based systems model, an adaptable mechanism for secure IPC, and the ability to remove potentially insecure parts of the kernel. Iit is also a fundamental security objective of the Linux kernel is to isolate user resources from one another.