

Draw It or Lose It

# **CS 230 Project Software Design**

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 | 02/05/23 | Kelvin Pichardo Doughty | Made updates in the Evaluation section in regards to Server Side, Client Side, Development Tools for each platform (Windos, Linux, Mac, Mobile) as well as response to questions in regards to each. |
| 1.1 | 2/23/23 | Kelvin Pichardo Doughty | Made updates to recommendations section Operating Platform, Operating Systems Arch, Memory Mangement, Distributed Systems and Networks as well as Security. |

**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 is looking to develop a web-based game called “Draw It or Lose It” that can run on multiple platforms that will allow it to be competitive on each one. The game is currently only available on the android platform. The purpose of this game is to have multiple teams consisting of several people going four rounds at a minute each. The application will have access to a library of images that will draw each image slowly allowing people to guess before the 30 sec mark expires. The other teams are giving 15 secs and have one opportunity to make a guess at the image if primary team does not.

## Requirements

* Needs one or more teams involved to play.
* Each team is made of more than 2++ players
* Game and Team names must be unique to allow users to check whether the name is in use or free
* Only one instance of the game can exist at any time.
* Must run on multiple platforms

## [Design Constraints](#_2et92p0)

These are the requirements that will guide the writing of the code and software for the game. While this is only the game aspect, we still need to look at application development. The Gaming Room would like this to run on all platforms. It is already available on android but need to work it into other mobile platforms. Along with machines like Windows, Linux, and Apple. To do this we will need to find a way to re-write the code in swift for the Apple platform or come up with a way to use existing code to be run on other devices by incorporating other languages.

## [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)

At the start of the UML Entity creates a relationship between Game, Team, and Player class. Meaning they inherit or get information from Entity. With UML listed below we can show this with inheritance. So, each class will share common references like “name” and “id”. Making Entity a superclass. When we look at their relationship, we see Team and Player is a “has a” type. While Game has a Team and GameService has Games. When we use UML, we call it aggregation (HAS-A). When a user “has a” it's an instance of one class and ties into another class. When we look at this diagram, we see GameService has a reference of Games, Games a reference of Tea, and Team a reference of 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](#_2o15spng8stw)

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Mac offers a server-based deployment method with the use of its built-in web server, Apache. It is a good choice for hosting web applications due to its stability and security. However, licensing costs for the server operating system may be higher compared to Linux. | Linux offers a server-based deployment method with various web servers such as Apache, Nginx, and Lighttpd. It is a popular choice for hosting web applications due to its stability, security, and low cost. Licensing costs for the server operating system are minimal or non-existent, depending on the distribution used. | Windows offers a server-based deployment method with its built-in web server, Internet Information Services (IIS). It is a good choice for hosting web applications, but may be more expensive compared to Linux due to licensing costs for the server operating system. | Mobile platforms do not typically offer a server-based deployment method, but rather a client-based deployment method through app stores. |
| **Client Side** | The application can be made compatible with all web browsers through the use of standard web technologies such as HTML, CSS, and JavaScript. Supporting multiple types of mobile devices may require additional testing and optimization. | The application can be made compatible with all web browsers through the use of standard web technologies such as HTML, CSS, and JavaScript. Supporting multiple types of mobile devices may require additional testing and optimization. | The application can be made compatible with all web browsers through the use of standard web technologies such as HTML, CSS, and JavaScript. Supporting multiple types of mobile devices may require additional testing and optimization. | Developing a responsive HTML interface for both Android and iOS platforms requires knowledge of platform-specific development tools such as Android Studio and Xcode. Supporting multiple types of mobile devices may require additional testing and optimization. |
| **Development Tools** | Popular programming languages used for building web applications on Mac include Python, PHP, and Ruby. Development tools include IDEs such as PyCharm, PHPStorm, and RubyMine, as well as text editors such as vim and emacs. There may be some licensing costs for these development tools, depending on the vendor. | Popular programming languages used for building web applications on Linux include Python, PHP, and Ruby. Development tools include IDEs such as PyCharm, PHPStorm, and RubyMine, as well as text editors such as vim and emacs. There may be some licensing costs for these development tools, depending on the vendor. | Popular programming languages used for building web applications on Windows include .NET, ASP.NET, and C#. Development tools include IDEs such as Visual Studio, and text editors such as Notepad++. There may be licensing costs for these development tools, depending on the vendor. | Development tools for mobile platforms include Android Studio and Xcode. There may be licensing costs for these development tools, depending on the vendor. |

## Recommendations Updated

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**: After analyzing the requirements of The Gaming Room, it is recommended to use Linux as the operating system for the server platform. Linux is a popular and reliable operating system that provides high performance, stability, and security. Moreover, it supports a wide range of hardware and software, making it an ideal choice for a game server platform. Additionally, Linux is open-source, which means that it is highly customizable and can be tailored to the specific needs of The Gaming Room.
2. **Operating Systems Architectures**: Linux operating system architecture is based on the monolithic kernel model, where the kernel handles all system operations and services. Linux also supports a modular design, which allows the system to be expanded with additional features and services as needed. The Linux file system is hierarchical, and all files and directories are organized under the root directory, which is denoted by a forward slash (/).
3. **Storage Management**: For the recommended Linux operating system, it is recommended to use the ext4 file system, which is a stable, high-performance file system. The ext4 file system supports journaling, which helps to prevent data loss in the event of a system crash or power failure. Moreover, ext4 supports large file sizes and file systems, making it an ideal choice for a game server platform.
4. **Memory Management**: Linux operating system uses several memory management techniques, including paging, swapping, and virtual memory. Paging is a technique where the operating system divides the physical memory into smaller fixed-size pages, and each page can be swapped in and out of the physical memory. Swapping is a technique where the operating system moves some parts of the memory to the disk, freeing up physical memory for other processes. Virtual memory is a technique where the operating system creates an illusion of a larger memory space than the actual physical memory.
5. **Distributed Systems and Networks**: To enable Draw It or Lose It to communicate between various platforms, it is recommended to use a distributed system architecture that includes a combination of client-server and peer-to-peer models. The client-server model will enable the centralized management of the game, while the peer-to-peer model will allow players to communicate with each other directly. The network architecture should be designed to provide high availability and reliability, and to minimize latency and bandwidth usage.
6. **Security**: To protect user information on and between various platforms, it is recommended to use secure communication protocols such as HTTPS, SSL, and TLS. The recommended Linux operating system supports various security features, including firewalls, access control, and intrusion detection and prevention systems. It is also recommended to encrypt sensitive user data using industry-standard encryption algorithms. Additionally, the system should be regularly updated and patched to address any security vulnerabilities.

**Conclusion:**

Overall, by adopting Linux as the server platform, using the ext4 file system for storage management, implementing memory management techniques, utilizing a distributed system architecture, and prioritizing security, The Gaming Room can successfully expand Draw It or Lose It to other computing environments while ensuring high performance, reliability, and security.

**Update:**

***SERVER SIDE:***

O Does each of the operating platforms offer a server-based deployment method where the website will be hosted?

* Linux offers a server-based deployment method with various web servers such as Apache, Nginx, and Lighttpd.
* Mac offers a server-based deployment method with the use of its built-in web server, Apache.
* Windows offers a server-based deployment method with its built-in web server, Internet Information Services (IIS).
* Mobile platforms do not typically offer a server-based deployment method, but rather a client-based deployment method through app stores.

O What are the potential licensing costs to the client, The Gaming Room, for the server operating system?

* For Linux, licensing costs for the server operating system are minimal or non-existent, depending on the distribution used.
* For Mac, licensing costs for the server operating system may be higher compared to Linux.
* For Windows, licensing costs for the server operating system may be more expensive compared to Linux.
* For mobile platforms, licensing costs for the server operating system are not applicable as they do not offer a server-based deployment method.

***CLIENT SIDE:***

O What is required of the application development process to ensure the application is compatible with all web browser platforms and mobile devices?

* To ensure compatibility with all web browser platforms, the application should be built using standard web technologies such as HTML, CSS, and JavaScript.
* To ensure compatibility with multiple types of mobile devices, additional testing and optimization may be required.

***DEVELOPMENT TOOLS:***

O What impact do these technical requirements have on a development team? Consider whether multiple development teams may be needed.

* The technical requirements for each operating platform and mobile device may require knowledge and expertise in different programming languages and development tools.
* This may require the use of multiple development teams, each specializing in a specific operating platform or mobile device.

O Are there licensing costs related to the development tools?

* There may be licensing costs for the development tools used, depending on the vendor. These costs should be considered when evaluating the overall cost of development.