

Draw It or Lose It

# **CS 230 Project Software Design**

Version 3.0

## Table of Contents

[**CS 230 Project Software Design** 1](#_Toc157086476)

[Table of Contents 2](#_Toc157086477)

[Document Revision History 2](#_Toc157086478)

[Executive Summary 3](#_Toc157086479)

[Requirements 3](#_Toc157086480)

[Design Constraints 3](#_Toc157086481)

[System Architecture View 3](#_Toc157086482)

[Domain Model 3](#_Toc157086483)

[Evaluation 4](#_Toc157086484)

[Recommendations 6](#_Toc157086485)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 01/28/2024 | Jordan Jenkins | Working game, team, and player classes  Added new class, entity class |
| 2.0 | 2/9/2024 | Jordan Jenkins | Completed Pros and Cons of Server- and Client-side for varying OS platforms. Accompanying developmental tools for each |
| 3.0 | 2/20/2024 | Jordan Jenkins | Providing recommendations to client on various systems. |

**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)

There are many challenges that people face when developing a *Win, Lose or Draw* style of game. One of the common challenges is that the players (users) were drawing images on an easel to assist team members with the puzzles. This is difficult to translate onto a web-based application where images need to be generated to assist with puzzles. Another challenge is the rise in demand of web-based application for Creative Technology Solutions (CTS) to tackle in which each new request is based on a different environment. The newest client, The Gaming Room, is requesting their *Draw It or Lost It* application to be deployed on a web-based platform. The game is currently deployed and available as an Android app only. The application renders images from a large library of stock drawings as clues rather than player drawings and would like this to be translated to a web-based environment.

## Requirements

The Gaming Room is requesting that the application can have one or more teams involved with each team having multiple players assigned to them. They are also requesting that each game and team name must be unique to allow users to check whether a name is in use when choosing a name. They only want one instance of the game to exist in memory at any given time. The Gaming Room believes that this should be attainable by given each game, team, and member a unique identifier.

## [Design Constraints](#_2et92p0)

When developing a web-based software, there are few design constraints to consider. One constraint is time where The Gaming Room is expecting the application to be finished by a specified date. Based on resources available and other projects being worked on, this might not be attainable. This goes hand in hand with developer’s workloads and personal skills. Another constraint will be dependent on the funding of the project. Depending on the amount of money that The Game Rooming is willing to invest in this project, might affect how polished the application appears. Another possible constraint is one of legality. With the expected unique identifiers of members of a team, comes privacy. There will need to be designs implemented to protect the privacy of players (users) which will also protect The Gaming Room and CTS from legal repercussions.

## [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)

There are seven classes in the UML diagram below, namely, ProgramDriver, SingletonTester, Entity, GameService, Game, Team, and Player. The Entity class is a parent class that relates to four child classes. The four child classes, GameService, Game, Team, and Player relate to each other in an association relationship where each class depends on each other. The SingletonTester class inherits from the

ProgramDriver class which is the main() program.

**"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** | **Characteristics**  Popular in web hosting, exclusivity  **Advantages**  Upgradeable, various options for different web hosting  **Disadvantages**  Less preferred for web hosting | **Characteristics**  Secured, highly preferred  **Advantages**  Security flaws are caught early before they become an issue  **Disadvantages**  Difficult to find application to support the web hosting needs | **Characteristics**  Widely used, dominant to other platforms  **Advantages**  Low loading times, high comfort ability  **Disadvantages**  High amount resources required, susceptibility to virus | **Characteristics**  Very popular, high portability  **Advantages**  Wide reach and compatibility, cost-effective  **Disadvantages**  Poor security |
| **Client Side** | Unique expertise from specialized support. Expensive to maintain and support and for the user. Ample time is needed to alter the software. | Unique expertise from specialized support. Not nearly as popular but is more optimized and therefore faster. | High expertise is needed since high resources requirements. Less optimization means slightly higher load times. | Relatively quick load times compared to other platforms. High popularity means there is higher technical support. |
| **Development Tools** | PHP programming language  JavaScript  Swift | PHP programming language  Perl programming language  C programming language | PHP programming language  JavaScript  HTML/CSS  C programming language | PHP programming language  C++  JavaScript  Java Programming |

## 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**: Based on the information provided, I think the Windows OS should be used. This environment is a common tool in developing web-based software. While the optimization might be less than Linux or MacOS, the load times are negligible. Its large availability makes it cheaper to maintain and relatively secure despite potential security vulnerabilities. The flexibility of Windows OS allows for support for new hardware while also being backwards compatible. It also has an ease of access that makes the operating system user friendly. With a friendlier GUI and plug-and-play ability, Windows seems like the best option.
2. **Operating Systems Architectures**: The layered design of Windows consists of two main components, user mode and kernel mode. It is a preemptive, reentrant multitasking operating system, which has been designed to work with uniprocessor and symmetrical multiprocessor (SMP)-based computers. Windows uses packet-drive I/O to process input/output (I/O) requests. The programs and subsystems in user mode are limited to what kind of system resources they have access to, while the kernel mode has unrestricted access to the system memory and external devices. The kernel mode also has full access to the hardware and system resources. The kernel is a hybrid kernel which means that the architecture compromises of simple kernel, hardware abstraction layer (HAL), drivers, and several services (Executive).
3. **Storage Management**: The recommended storage management for this application would be database management system (DBMS). Due to high compatibility with Windows OS and its adaptability to other platforms, this would be the best storage management to develop the web-based application with potential to adapt it to other platforms in the future.
4. **Memory Management**: Windows manages memory by using memory compression. This process reduces the size of inactive data in the random-access memory (RAM) to free up unused space and allow more programs to run at once. This would allow the *Draw It or Lose It* game to run more efficiently. This memory management also improves system performance. Each process on 32-bit Windows has its own virtual address space that enables addressing up to 4 gigabytes of memory. Each process on 64-bit Windows has a virtual address space of 8 terabytes. All threads of a process can access its virtual address space. However, threads cannot access memory that belongs to another process, which protects a process from being corrupted by another process.
5. **Distributed Systems and Networks**: Using LAN as the networking technology, the distributed system will use hubs to connect multiple computers such that when one computer crashes, the game still operates. The computers can be set up as redundancy to take the load of the downed computers until they are fixed. The hub will also serve as a repeater to amplify the signals that deteriorates when travelling for a long distance. Having LAN as the reliable network and hub as the connectivity hardware, it will help reduce the likelihood of an outage.
6. **Security**: Windows firewall and security measures will allow for security amongst users. The encryption of user data will protect against outside threats. Windows Security continually scans for malware (malicious software), viruses, and security threats.