

The Gaming Room

# **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 | 07/15/2024 | Clayton Merritt | Analysis of development for server and client sides of the application. |

**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) - *Write a summary to introduce the software design problem and present a solution. Be sure to provide the client with any critical information they must know in order to proceed with the process you are proposing.*

The Game Room (TGR) has approached our company to assist in developing a new game that’s based on their current game, Draw it or Lose it. Since it's already been developed on Android, they have an idea of how it should function already. However, they really don't know how to set up the web-based environment, which is where we come in. What ever software we end up developing will determine what type of hardware is required for the program. They have provided a document, which is attached, that gives us an idea of how its expected to function, although this “entity” class needs to be clarified within the UML Diagram.

## 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*

* Games will have the ability to have one or more teams involved.
* Each team will have multiple players assigned to it. Something that needs to be clarified here is how do we deal with players that aren’t on teams? Will they get auto assigned to a team?
* 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.
* Game Must be compatible with all platforms

## [Design Constraints](#_2et92p0) - Identify the design constraints for developing the game application in a web-based distributed environment and explain the implications of the design constraints on application development.

* TGR not entertaining hardware requirements of other platforms while software is being developed. Hardware Requirements will be an issue regardless because different platforms have their own requirements to function.
* TGR not being specific about what is different from Draw it or Lose it? We know how their other game works. Will this new game function the same way?
* Having game developed in an existing language and using it as a template may raise expectations of an outcome.
* Not every platform is going to be compatible with what is developed. It may be worth it to make it compatible with major platforms and then go from there.

## [System Architecture View](#_ilbxbyevv6b6) //NOT REQUIRED

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) - Describe the UML class diagram provided below. Explain how the classes relate to each other. Identify any object-oriented programming principles that are demonstrated in the diagram and how they are used to fulfill the software requirements efficiently.

So the provided UML class diagram gives us an overview of how TGR’s new Game architecture should operate. It has a driver class which uses the Singleton tester to see whether or not the game service actually is Singleton. Beyond that, you can see an entity class which is a superclass. It allows the game, team and player subclasses to inherit an Id and a name. This, generally is useful, especially if you have different subclasses that will have the same attributes like a name and an id. If you go further beyond the entity superclass, each subclass has their own individual lists so that games can store teams teams can store players all while inheriting an ID and a name from their superclass in addition to methods to get each variable.

**"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 -**Evaluate Mac / Linux / Mobile devices for their characteristics, advantages, and weaknesses for hosting a web-based software application. | Limited support for server-side capabilities within Mac. Limited Terminal options, limited support for hardware and processing power. Furthermore, Mac *does not actually have a server platform to host a website on.* | For server-side options, Linux appears to be a powerhouse. It supports thousands of processors, and depending on whether you’re using 32-64 bit architecture, millions of terabytes of ram and a Great Terminal for debugging.  Linux, depending on OS Build (IE Red hat vs Ubuntu), is free and is the most popular OS to use, as far as servers go. | Windows PCs are not quite the powerhouse that Linux is. It supports 6 terabytes of ram, 2-4 processors. Though it comes with command prompt terminal, its not the greatest. It does support multiple virtual boxes giving it the ability to be flexible and allow for the development and testing of the game on several platforms. Unfortunately, windows server isn’t free. | Limited processing power and system memory available to host web based applications for many people to use make this the least desirable platform to host a server on. These are more streamlined to be client platforms as they have no terminals to interpret what is running. Besides, who hosts a website on a phone or iPad? |
| **Client Side-** Determine the software development considerations (cost, time, expertise) that are necessary for supporting multiple types of clients as they pertain to Mac / Linux / mobile device. | Upon further reflection on this topic, it appears to me that considerations for client-side development of this game should be addressed as a whole and not on an OS-by-OS basis: The most popular web browsers, i.e. Chrome, Safari, and Edge, can be found across Mac, Linux, Windows and even mobile devices making up nearly 90% of the market share of web browsers currently in use today (Vailshery, 2024). From a business perspective, focusing on the development of Draw it or Lose it across these clients allows The Gaming Room (TGR) to have their game accessible to the vast majority of users across the internet first. If it’s a paid game, It’ll allow TGR to recoup the expenses incurred by development and testing and also prepare them for the continued development and testing for browsers that are less common, which may or may not be worth developing for due to cost-benefit reasons. But this is something that we need to run by TGR as it is not our choice as to what they do with their time, money and resources. But as the consulting firm they are working with, we should bring it up. | | | |
| **Development Tools -** Identify the relevant programming languages and tools (IDEs and other tools) that are used to build this type of software for deploying on Mac / Linux / Mobile devices | Most popular languages for Mac in include Java, JavaScript Python and Swift. Dev tools include but aren’t limited to Eclipse, Xcode and Atom. | Most popular languages for Linux include: Python, C and C++, Java and Java script. Dev tools include but aren’t limited to VS Code, Eclipse, git, and Atom. | Languages for Windows include C#, Visual basic, Javascript and Java, python etc… There are several dev tools available for windows: VS code, eclipse are prime examples. | Primary language for programming in android is Java and Swift / objective-c for iOS. There are several dev tools available for developing for mobile environments: Xcode and Eclipse |
| NOTE: Its Important to note that these are just for the development of code on the OS in question for *the game only*, not for the creation of the art, graphics, and sound of the game. The requirements of this need to be further discussed with The Gaming Room. | | | | |

## 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**: Generally speaking, Windows and Linux appear to function in a similar fashion both with their storage and memory management. As a consultant, what it really boils down to is cost. If Cost is no object, I’d recommend developing this on a Windows operating platform. Draw It or Lose it does not appear to require a powerhouse of an operating platform to function. There are several development IDEs on Windows as well as tools for its development. Plus, it has the ability to emulate other operating systems as required to test its functionality before deploying it to other operating systems. If sustainment cost is an object, Linux would be a better choice as it’s a free (server) operating system. I will continue with this under the pretext that cost is no object, developing for and on a Windows based server operating system.
2. **Operating Systems Architectures**: Windows is composed of two basic parts: the user part, which focuses on what the user is doing, and the kernel part, which focuses on system resource allocation based on what the operating system and users need. The user makes a request, the kernel acts as a middleman between the actual system hardware and the user. It then allocates required resources for the user’s request. Both Windows and Linux Operate in this fashion.
3. **Storage Management**: For the recommended storage system, a block type storage management system seems to be a good fit for what TGR needs for Draw it or Lose it because of how it handles storage. It appears to be flexible and it compartmentalizes who has access to read and writing abilities to the storage system, which is what I think TGR wants in their program, although a file storage management system seems equally as appropriate as a block storage system. Object type storage management solution doesn’t really seem to be as important here because there doesn’t seem to be a multitude of different data types to be stored: Some type of JSON to store users data and some type of basic storage system to store photos in.
4. **Memory Management**: Windows uses Virtual Memory, paging and paging tables memory management. Its paging tables are stored in the processors memory management unit (MMU), or in physical memory where it waits for a call from the actively executing program and retrieval for execution. This, however, appears to be standard across operating systems.
5. **Distributed Systems and Networks**: Ultimately, the network that will connect Draw it or Lose it between users will be the internet using HTTP protocol (ReST Environment in addition to Java) which appears to be universal among most operating system web browsers. People using the game connect to some type of game server which ensures users have the permissions to use the game and then It synchronizes its players to its own clock so that they all have the same time allotments. Game outages might be interesting to deal with here, However, using a windows server platform doesn’t necessarily mean that you have a singular windows based server that multiple users connect to to play the game. TGR has options. It could utilize Azure and host as many servers as required for the game load or they could invest in a physical server. This, I feel, is where Windows outshines Linux.
6. **Security**: Security can refer to several different things: it can refer to the isolation of users from one another, it can refer to the protection of vital files from malicious exposure, it can also refer to the detection of malicious code being run on the operating system. The good news in this department is that Windows has several security options, from firewalls to anti-malware and virus detection and isolation software available. For a premium.

# Bibliography

Vailshery, L. S. (2024, June 21). *Market share held by leading internet browsers in the United States from January 2015 to May 2024*. Retrieved from statista: https://www.statista.com/statistics/545520/market-share-of-internet-browsers-usa/#:~:text=U.S.%20market%20share%20held%20by%20internet%20browsers%202015-2024&text=As%20of%20May%202024%2C%20Google,was%20released%20in%20January%202020.