

<Draw it or lose it>

# **CS 230 Project Software Design Template**

Version 3.0

## 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) **& 4**

**Evaluation 4 & 5 & 6**

**Recommendations 7 & 8**

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 3.0 | 3/3/2024 | Abdulrahman Al-Nachar | This is the completed final version. |

## [Executive Summary](#_sbfa50wo7nsh)

The client has a game *Draw it or lose it* that’s only available on android devices and wants to make his game available on PC across all platforms and operating systems.

## Requirements

1. *Create a game that is based on his current game, maintaining the same level of consistency and quality used.*
2. *Develop a web-based app for the game.*
3. *Make that game accessible on PC regardless of the OS or platform used.*

## [Design Constraints](#_2et92p0)

1. Develop the web-app in a language that is platform independent to make it the app cross platform.
2. Design the app to handle a growing number of users, since the app is cross-platform, we should expect a bigger number of users compared to the Android app to support potential scaling.
3. Make sure that the client’s objectives and goals are met without exceeding his budget for the app while maintaining quality and functionality and achieving his stated objectives.

## [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)

In this diagram, it shows three things:

1- The Entity class is a superclass with 3 subclasses (Game, Team, Player), So they’re seen as a type of the Entity superclass, and this relationship is called inheritance, those three classes get to use the attributes and functions of the superclass for their functions.

2- There are three association relationships between 4 classes with the ‘0...\*’ notation, this relationship means that every instance in class #1 is associated with 0 or more instances in class #2 while all instances in class # 2 are associated with one instance in class #1.

A- Team and player, The Team class’s instances are associated with 0 or more instances in the Player class while all the instances in the Player class are associated with one instance in the Team’s class.

B- Game and Team, The Game class’s instances are associated with 0 or more instances in the Team class while all the instances in the Team class are associated with one instance in the Game’s class.

C- GameService and Game, The GameService class’s instances are associated with 0 or more instances in the Game class while all the instances in the Game class are associated with one instance in the GameService’s class.

3- The main function that runs the program is in ProgramDriver and it has a dependency relationship with the SingletonTester Class, meaning that ProgramDriver depends on SingletonTester class’s function for its implementation and uses SingletonTester in some capacity.

Some of the OOP (Object Oriented Principles) used in this UML , is Inheritance between the Entity superclass and its three subclasses , The singleton pattern being used in GameService. and they’re used to achieve the program’s goal in an efficient way like the Singleton pattern to make sure that one instance of GameService can exist in memory at one time.

**"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 used to offer server-based deployment using MAC OS X server but it has been discontinued since 2022, however you can still deploy applications using alternative web server software that would work on mac like Apache, and that doesn’t have any licensing costs  **Characteristics**:  Mac has a friendly UI.  **Advantages:**  Mac is less vulnerable to viruses and malware; mac offers integration with other mac products like the iPhone and the iPad.  **Weaknesses:**  Mac’s biggest weakness is that it offers limited compatibility outside its own products. | Linux is open source which means that there are no licensing costs associated with using it.  **Characteristics**:  It’s free to use with no licensing costs and Linux is stable and highly reliable.  **Advantages:**  It has good performance and less loading times, it’s secure since it is equipped with several features that make it hard to breach.  **Weaknesses:**  It does require a learning curve for administrators and developers, and it has poor tech support. | Windows is a bit more expensive than the others. It has licensing costs however it does offer server deployment services through its own Windows Deployment Services.  **Characteristics**:  It integrates with Microsoft products and is user-friendly and simple to use.  **Advantages:**  It’s widely used and is common, its hardware is usually cheaper than Mac thus making it widely used.  **Weaknesses:**  It costs more than others, It does have licensing costs that make it more expensive and the cost is dependent on the type of package you need. | Mobile devices can't host apps on their own, so the app is hosted on a different platform or using the cloud.  **Characteristics**:  Limited resources, mobiles don’t have CPU power like a laptop, it has less memory, less RAM, etc., so it makes it limited in resources.  **Advantages:**  Scalability, Mobile devices are available everywhere and open the way to get customers that don’t have any other devices there are more phone users than computer users.  **Weaknesses:**  Since its limited resources will affect its overall performance, a game on mobile devices is designed with less detail compared to its web app and that is due to mobile device’s limited resources. |
| **Client Side** | The cost would be for the hardware and potential cross-platform tests, the time it takes for mac is the average time needed, we don’t need extra time when using mac, we need expertise in web development all languages do work in mac , but the preferred one for mac would be swift since its integrated with all of mac’s products and platforms. | The cost would be for the hardware and potential cross-platform tests, the time it takes for Linux is higher than the average time needed, Extra time is needed for the developer to learn how to use it and get around the learning curve associated with it.  Developer would need to implement responsive web design to make sure that it integrates to other platforms and systems | It wouldn’t cost more on Windows; it will take some extra time for developers using Windows to learn Windows-specific tech like their APIs and languages like C# and .NET and developers would be required to know the languages needed to design responsive apps that are functional cross-platform. And knowing IOS swift coding language and Java for mobile devices. | It would cost more in license fees, it takes more time since usually developers are used to do this on a laptop, and developers would need to have the expertise to build an app the is cross-platform and that includes knowing web development languages and knowing like HTML & CSS & JavaScript as well knowing the frameworks needed. And knowing IOS swift coding language and Java for mobile devices. |
| **Development Tools** | The languages needed would be the ones for web development like HTML, CSS, and JavaScript, as well as Swift, mac has its IDE named Xcode which supports a lot of coding languages and their libraries. | The languages needed would be C/C++ and JavaScript, visual studio code and eclipse are available on Linux and there are no extra costs to them since they are open-sourced. | Visual Studio is the IDE mostly used on Windows and it is free and there are no extra costs associated with using it. | Swift & and Java would be two important ones to know and then the languages for the web app, Xcode is an IDE that works here as well as some other IDEs like VSC, IDEs are typically free to download however frameworks and hosting services do require fees or licensing fees. |

## 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 would recommend Windows Server platform for the game, Albeit Windows can be a bit more expensive overall than other servers it is more reliable, and it is built-in to all the devices and platforms within the Windows eco system and at the same time supports creating an app that is cross-platform with its .NET framework and it’s easy to use IDEs.
2. **Operating Systems Architectures**: Windows has several core systems in its architecture and the main ones are:
3. Drivers: Drivers is a software that delegates tasks and helps communication between the two different modes and it acts as a layer of protection and abstraction so that it protects the system from malicious code.
4. Kernel mode: it's the center of its architecture, and it controls everything within, it manages the hardware and ensures everything runs smoothly, it is responsible for running operating system functions.
5. User mode: User mode runs the system functions for applications but not core CPU functions.

1. **Storage Management**: Windows has Storage systems that can help us with The Game Room.
2. Storage Spaces: Storage Spaces is a Windows technology that helps by combining physical and virtual disks to protect the system from driver’s fails and improve performance.
3. NTFS: NTFS (New Technology Files System) is a Windows built-in default system to manage storage, it enjoys multiple features like compression, encryption, and managing permissions.
4. **Memory Management**: there are several parts to managing memory in Windows:
5. Virtual Memory: this is a technique to effectively manage memory specifically for multitasking, it manages the physical memory of the computer’s architecture so it is responsible for allocating physical memory to different functions and operations, and it allows for operations to have its own virtual space so that it keeps the memory from being overloaded.
6. Memory Paging: this technique gets the computer to store and call data from secondary storage to be used on its main memory. It helps with improving the computer’s performance by letting programs exceed the size of the computer’s physical available memory.
7. Memory Protection: This technique protects the computer from bugs and memory by allowing processes to access memory that is allocated to it and preventing processes from accessing memory not allocated to it.
8. **Distributed Systems and Networks**: For devices and applications on different platforms to communicate with each other, Windows Server OS does support using RESTful API to do that and communicate cross-platform on the web application through the internet using HTTP, Windows’s ASP.NET Framework makes it easy to build HTTP Services that will be used by users on different devices and platforms and supports building RESTful applications on it.
9. **Security**: Security is a fundamental part of the application and there are several techniques to apply to ensure security in The Gaming Room Win it or Lose it.
10. Data Encryption: Encrypting data transmitted in the app through different platforms using SSL protocols over the internet.
11. Access control and permissions: Implement mechanisms to restrict unauthorized access and limit users' permission to what they need in the application and utilize user verification methods to prevent users from being hacked like passwords & unique usernames and multi-step authentication.
12. Firewalls: Firewalls are network security devices that come in software & hardware, and It's responsible for preventing malicious software over the internet from accessing the network or end user’s computer.