

<Name-of-Software-Application>

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template**](#_l6ti7uoag22u)1

[**Table of Contents**](#_30j0zll)2

[**Document Revision History**](#_grjogdjh5fi8)2

[**Executive Summary**](#_sbfa50wo7nsh)3

[**Design Constraints**](#_2et92p0)3

[**System Architecture View**](#_ilbxbyevv6b6)3

[**Domain Model**](#_8h2ehzxfam4o)3

[**Evaluation**](#_2o15spng8stw)3

[**Recommendations**](#_m8aleynsvzvc)5

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 6/19/2022 | Hanna Rawson | <Brief description of changes in this revision> |

**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)

We are a technology consulting firm, Creative Technology Solutions, and our client, The Gaming Room wants to develop a web-based game that serves multiple platforms based on their current game, Draw It or Lose it. Their current game is only available on Android app. The web-based game must be capable of supporting more than one team. The game and the teams’ name must be unique, as only one game can exist in memory at any given time.

## [Design Constraints](#_2et92p0)

* The game needs to have the ability to have one or more teams involved
* Each team will have multiple players assigned to it
* The game and the team’s name need to be unique. This will allow users to check whether a name is in use.
* Only one instance of the game can exist in memory at any given time. This will be accomplished by creating a unique identifier for each instance of a game, team, or player.

## [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)

Looking at the UML diagram below we can see that the Entity class is a collection of attributes. The entity class is also a parent class to Game, Team, and Player. In other words, it’s also an inheritance, this is because we can see a white arrow.

All the classes Game, Team, Player, Game and Gameservice have the same multiplicity. The multiplicity give you an information about how many instances of specific type, attribute values or connected instances can be created. In this case, all the classes have zero to unlimited instances. ProgramDriver class is the main method, which then calls for the SingeltonTester. The SingeltonTester will test the code. The Singelton’s purpose is to control object creation, limiting the number of objects to only one.

**"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** | Mac OS X Server was provided as the OS for the Xserve computers, rack mounted server computers designed by Apple. Mac is not as popular as Windows, so need a user that is familiar with it. Server is cheap. | Linuxserver refers to a specific type of server based on the free, open-source, and modular Linux operating system. Since Linux is an open source, the cost is low. Linux is less common than Windows and Mac, so users need to be somewhat familiar with the server. | Windows does offer its own server, Windows server. Windows is the most used OS and most users have no trouble navigating through it. | Hosting a server on a mobile device may not be as successful as doing so on a computer. A mobile device does not have the same capabilities. It is possible to host a server on a phone, but it’s unlikely to be usable without a special arraignment from the service provider. The phone sits with one or more firewalls and NAT devices between it and the internet. |
| **Client Side** | Mac is not an open source, so the cost will be higher. I think Mac is less popular than Windows, but more popular than Linux. Need to find someone comfortable and experienced enough, so time would depend on the expertise. | Linux is open source, so the cost is low. I find Linux to be less common and that’s why knowledge can be hard to find. Need to find a someone who is experienced in Linux, so they can take their time studying it. | Windows is not an open source, so the cost here could be higher. Time could either take less or longer time, it depends on the experience an individual has. If they are experiences, less time is needed. If they aren’t as experienced, it may take them longer. | Because we are using a mobile device, the cost would be lower. This may take some time, because we are working on a mobile device - Mobile devices are also easier to work on than others, so experience-wise this should not be any issues. |
| **Development Tools** | The most common language used to write in any Mac apps is Swift. Mac can also be used by IDE like Atom. | Eclipse is popular development tool pertaining Linux. Atom is also another popular choice, it’s free and open-source text. Atom comes with support for plug ins written in JavaScript and embedded Git Control. | For Windows the development tools would be Visual Studio or Eclipse, they are the most popular ones. Can be used to develop C#, HTLM, JavaScript as well. | Just like development tools for any iOS apps and Mac, the iPhone has the same development tools. Majority of the iPhone apps are written in Swift. |

## 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**: Well in the beginning, our client, The Gaming Room wanted to develop a web-based game that serves multiple platforms based on their current game, Draw It or Lose it. I would choose Windows. Windows will allow Gaming Room to expand to other computing environments, more available software, cost may be higher but minimum expertise is required.
2. **Operating Systems Architectures**: The architectures of Windows, is a layered design. Consists of two main components, user mode and kernel. The Windows kernel serves as the core of an operating system and the interface between software and hardware. The kernel is in constant use and is a key component of the operating system.
3. **Storage Management**: Using a SSD is the best for a video games. The SSD speeds uploading times and launch times. Windows 10 also has something called storage sense. It works as a silent assistant on our behalf with little no so setup. Windows has something called Windows Disk Management and is a built-in disk management program, called Disk Management. It can help us manage hard disk partitions without rebooting the system and any interruption.
4. **Memory Management**: With this being a game, we would need to equip the database with plenty of photos/pictures. Memory Management is an essential function of the OS. Microsoft Windows has its own virtual address space for each 32-bit process, allowing up to 4 gigabytes of memory to be viewed. Each process has 8-terabyte address space on a 64-bit Windows.
5. **Distributed Systems and Networks**: Distributed systems must have a network that connect all components (machines, hardware, or software) together so they can transfer messages to communicate to each other. You need to make sure no outages are happening and need to make sure the servers can take the load.
6. **Security**: Managing identities is the first step in protecting your environment. To ensure security, we must protect user data and the system as well. To keep client data secure, VPN could be used. A virtual private network, protects your identity and browsing activity from hackers etc. A VPN gives you online privacy and anonymity. All data should be encrypted.