

Draw It or Lose it

# **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 | 01-26-22 | Devon McCain | Beginning design of Software Design Template |

## [Executive Summary](#_sbfa50wo7nsh)

The problem that the client is having is that they are creating a web-based game application, but they do not know how to set up the environment for it. The client will need to discuss which programming language they would like to use along with a website that they plan on linking the game to. The way we can solve their problems is by setting up the environment for the client and discussing a proper fit for the programming language, along with getting a website ready for the implementation of the game.

## [Design Constraints](#_2et92p0)

There are some constraints that should be noted. One of them is the programming language that the client would find preferable to use. After this, they will need the game to have the ability to have one or more teams involved. Each team should allow 2+ players to join, along with the game and team names need to be unique. Finally, we need to make sure that only one instance of that game can exist within the memory at any given time.

## [Domain Model](#_8h2ehzxfam4o)

Team, Player, and Game all have a “is a” relationship with Entity. This means that each of these are an Entity, and they are all inherited by Entity. All three have similar attributes of ID and Name. The relationship between Player and Team is a “has a” relationship. Team has the attribute Players and similarly, Game has Teams and GamesService has Games. This can be defined as Aggregation in UML format. The “has a” relationship means that an instance of one class has a reference of an instance from another class. GameService also has reference of Games, while games has a reference of Team, and finally Team has the reference of Player. GameService can have multiple games, although each Game can have multiple Teams, and each Team and have multiple Players.

**"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** | **Advantages:**  Mostly gain advantages if there are Mac clients within the network.  Allows for easy creation of features.  **Disadvantages:**  Uses a small set of hardware that can limit your hardware options.  Some productivity software is not available on Mac. | **Advantages:**  Known for their stability and reliability.  Security  **Disadvantages:**  Certain software not available to Linux.  Would require training staff in usage if they have no/little experience with Linux. | **Advantages:**  Supports traditional hypertext mark-up.  Runs similarly to Windows  Software compatibility  **Disadvantages:**  More expensive  Less stable  Not as secure as Linux. | **Advantages:**  Can be inexpensive if you have the expertise for the backend coding  Disaster recovery in real-time  **Disadvantages:**  Most difficult option, especially when it comes to backend coding.  Must find a provider to host the app  Requires multiple skill sets  Bandwidth limitations |
| **Client Side** | Costlier than Windows and the user is then forced to use a Mac device to run the program. | No upfront costs for the software, but it does require more time for training to gain an understanding of it if there is no one with this understanding. There would also be the need of a Linux compatible computer. | Great option with multiple platforms to write code with. An easier userbase to access as well with Windows being very popular. Wouldn’t require as much time with learning, but there is an upfront cost. | There must be development for multiple different form factors. This can lead to different designs within the application. The UI will have to be designed specifically to be more accessible on the smaller form factors. |
| **Development Tools** | The most common programming language for Mac is Objective-C | Supports multiple different types of programming languages and has multiple IDEs as well. | There are many programming languages with Windows as well. Although, C++ is the most used for the core of applications.  Visual studio and Eclipse are good options for IDEs. | There are two different languages that may need to be used.  For IOS, Swiftic is one of the best app development frameworks.  As for Android, Java is the official language for their app development. |

## 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**: There are multiple different platforms that Gaming Room could use. However, there are many differences between all of them. The best option would be Windows, as it is reputable and gives access to thousands of users from day one. Along with this, if Java or C++ is used to code it, it would make the transition to mobile gaming on Android relatively easy besides formatting issues.

**Operating Systems Architectures**:

X86: Huge developments have been made to easily develop games on Unreal or Unity. Large hardware compatibility, support, and very powerful.

ARM: Has many limitations and does not compare to the power of X86. Could be a possibility for mobile gaming.

1. **Storage Management**: HDD is an option, although it is terribly slow compared to SSDs that many people are using. Although more expensive per gigabyte, but worth it in speed and future proofing. SSD would be the best option, along with it having the supportive options of NVME and SATA.
2. **Memory Management**: Every process on a 64-bit Windows has a virtual address space of around 8 terabytes. Although, threads cannot access memory that belongs to another access which can cause corruption via another process.
3. **Distributed Systems and Networks**: This can be accomplished by creating a network that is a multi-user interaction system that most network games typically use. This will allow a database that is shared among the players to interact with each player. The biggest downfall of this option is that this will most likely need to be done from scratch. There is the option of Artery, which is designed to support network gaming applications. It provides an interface, which is high level, and it takes advantage of application semantics to better optimize the network performance.
4. **Security**: A cross-platform security system will be needed unless each individual operating system gets their own. User information can be protected by using a complex encryption for data that is transmitting from user to client. There will also be need for the information that the client must be protected as well. VPN’s and other encryptions for data will be needed here as well.