

<Name-of-Software-Application>

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[CS 230 Project Software Design Template 1](#_Toc664784576)

[Table of Contents 1](#_Toc1759185264)

[Document Revision History 2](#_Toc2058254750)

[Executive Summary 2](#_Toc1493574312)

[Requirements 3](#_Toc136035518)

[Design Constraints 3](#_Toc1922333471)

[System Architecture View 3](#_Toc1397262338)

[Domain Model 3](#_Toc987777702)

[Evaluation 4](#_Toc1665687180)

[Recommendations 5](#_Toc252097193)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 11/26/23 | Brandon Thaller | Updated all Questions and formatted table of contents. |

**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)

The Gaming Room wants me to develop a web-based game that is based on the game draw it or lose it. The Gaming Room has requested my help in streamlining the development and developing the game application.

## Requirements

* A game will have the ability to have one or more teams involved.
* Each team will have multiple players assigned to it.
* 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.

## [Design Constraints](#_2et92p0)

* The game must run on multiple platforms
* Each game will have the ability to have one or more teams involved
* The application will render imaged from a large library of stock images
* Only one game instance can exist in memory at a time.

## [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)

The Game, Player and the Team class are in a relationship with the Entity class inheriting the information from it. The ProgramDriver class is the class running the package and uses the SIngletonTester class when running. GameService, Game, Team, and Player are associated with eachother.

**"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** | Macs have an easy to use interface, with flexible terminal commands. They are typically great for servers due to their simplicity. | Linux has flexible terminal commands and are typically more difficult to use. They are great for servers due to being very open source. | Windows has an easy to use interface and is more widely used for everything. Terminal commands can be difficult due to needing lots of steps. They are good for servers but tend to be lackluster. | Mobile devices have a very easy to use interface, but I imagine their use as a server would be limited due to hardware constraints and use. |
| **Client Side** | The cost of mac tends to be more expensive than windows and linux. If you're not as familiar there could be a learning curve making them more expensive and time consuming. | Linux hardware is extremely cheap but if you're not familiar with Linux the learning curve can be steep being more time consuming. | Windows is extremely easy to setup and can be cheap. Windows servers are very plug and play with very little learning needed making them the best time wise. | I imagine mobile devices don’t make good servers, although they are easy to use they lack good hardware and they tend to be pretty expensive. |
| **Development Tools** | Languages on Mac consists of HTML, CSS, C++, Python, and JavaScript, Pycharm and Visual studio are good IDE’s but some commands in C++ don’t work right on Mac. | Linux can use most of what Mac can but also support other languages like Ruby, and PHP. | Windows is the best OS for programming supporting most languages to date like C++, C, C#, Python, Java, and many others. | On mobile devices I know you can compile Java, and Python, and C++. Most backend languages are supported. |

## Recommendations

Analyze the characteristics of and techniques specific to various systems architectures and make a recommendation to The Gaming Room. Specifically, address the following:

**Operating Platform**: I think Windows is the best operating system to expand Draw It or Lose it to other systems. Windows tends to be the best in terms of flexibility compared to other systems when it comes to available hardware, costs, and user ability.

**Operating Systems Architectures**: Windows is extremely easy to use and tends to be on the cheaper side. There is few IDEs that won't work with Windows and few many that are not outright written in a Windows environment.

1. **Storage Management**: Windows has a great storage system, being able to have drives in a RAID configuration and with the preinstalled software to manage the drives makes it extreme simple for managing storage.

**Memory Management:** Windows storage strategies make it possible to load files to RAM and depending on the program could even use web hosting to call these files and save them later to maximize preformance.

**Distributed Systems and Networks**: Windows is widely used that outages and internet connectivity is easy to diagnose and troubleshoot. Being that it is so widely used makes it more user friendly than other systems.

**Security**: Windows has preinstalled security features like Windows Defender. Securing user data may be done server side and would best be done with foreign keys.