

<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 | <06/01/21> | <Hendell Toussaint> | <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)

The application consists of a four-round guessing game. One minute is allotted for each round. The game is inspired in part by the 1980s game Win, Lose, or Draw, in which teams predict what will be sketched. The Client wishes for their game to be larger and superior than the game on which it is based, a 1980s game called Win, Lose, or Draw. The client also requests that the game enable multiplayer and that each game have unique names for individuals and teams, prohibiting others from using the same names.

## [Design Constraints](#_2et92p0)

The main design constraint I see is that customers must be able to play Draw it or Lose it on more than one operating system, not only Android. More programmers will be needed to create this code for numerous operating systems. The Client also requires the game to have names that are unique to players and teams, preventing others from using the same names.

## [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 Entity class is related to the Game, Team, and Player classes, all of which are Entities. All three are derived from Entity, which is depicted in UML thru inheritance. This is known as a "is a" relationship. This is a super class Entity since all three have common characteristics in terms of id and name. The relationship shown below between the team and the player is called a “has a” relationship. Players are found in teams, Game has teams, and GameService arranges Games. We can see this depicted below. This is called Aggregation. The relationship term "has a" can be defined as an example from one particular class that refers to an occasion from another class.



## [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** | <Although it is feasible to run a server on MAC OS, it is a less common choice. One significant disadvantage of Mac OS is that in order to get the most out of it, you will need to work with someone who is familiar with the operating system. The server will be more specialized to get the most out of the application on Mac OS, which is a benefit. The cost is reasonable, but having someone devoted to administering the server may raise pricing as well as the license fees for the operating system.> | <Linux is the most common operating system used to run a server. the initial investment is modest and manageable because it is an open source operating system. To get the most out of it, like with Mac OS, you'd need a skilled server administrator. Because Linux does not require any licensing payments, the cost is lower.> | <Between license costs and setup, Windows OS would be the most expensive first expenditure. The appeal of Windows is its usability. Windows is the most well-known operating system, and someone should be able to operate it without being specialized. Windows also comes with a slew of built-in apps that may help you get the most out of your server. > | <When compared to the power of a dedicated computer, running a server on a mobile device makes little sense. The restrictions would not be in terms of money, but rather of technological capability.> |
| **Client Side** | <Mac is a client who requires a lot of labor. To confirm that the coding was compatible, someone knowledgeable about the operating system would need to be there. To ensure cross-platform compatibility, it is preferable to make the program as basic as possible.> | Because Linux is not a widely used platform, it would involve the presence of an expert. Development expenses should not rise, but the team of people who can work on the project should be reduced to save time and money.> | <In theory, Windows should take less time. It is a commonly used system that the team should be familiar with. There are several programs that can help with development, and the team should use these tools to reduce the time required to build the app.> | <Mobile devices are commonly used objects, and they should not have a detrimental influence on development time. To ensure optimal operation on all operating systems, the team should prioritize testing and verifying that everything works at all phases of development.> |
| **Development Tools** | <The most prevalent programming languages and IDEs deployed on Mac OS, like many other OS system configurations, are JavaScript, CSS, Python, HTML, and others. Tools involve libraries for several languages, Eclipse, Visual Studio, and so forth.> | <Ruby on Rails, Java, Python, CSS, JavaScript, HTML, and other common languages are utilized. Nodejs, Visual Studio, Github, Repl.it, and command prompt are examples of development tools.> | <Ruby on Rails, Java, Python, C++, C#, JavaScript, HTML, among others are the customary languages used. Visual Studio, Eclipse, Repl.it, command prompt  are development tools> | <The most common languages include Python, Java, CSS, JavaScript, HTML, php, rails ruby, C++, and more. Includes technologies like Repl.it, Nodejs, Github, Visual Studio and the prompt command (power shell).> |

## 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**: <Taking everything into consideration, I believe the best Operating Platform will be a Linux OS. This conclusion is based on the fact that Linux is a widely used open-source server operating system. Being open-sourced is generally cheaper and works well with the other solutions we should pursue. Dedicated server expertise can be delegated to a cloud provider, further lowering costs. Linux also reinforces the application's capacity to operate on all platforms.>
2. **Operating Systems Architectures**: <The Linux architecture is built around the Linux Kernel, which houses all of the operating system's core functions. In addition to the Kernel, there are various layers of the system, including the shell, which serves as the interface between the user and the Kernel. This configuration allows us to be flexible in creating the best shell for our needs for this application.>
3. **Storage Management**: <A cloud computing provider will provide the best solution for storage management. The Google Cloud Storage , in my opinion, is one of the best provider for this project. This storage would deliver the best results at a very low cost. Google Cloud Storage also provides a variety of Linux Operating System options that will meet the needs of the Application. Another advantage of using third-party storage is that scalability is significantly less expensive.>
4. **Memory Management**: <Memory management will be handled efficiently by the Linux operating system. Linux uses virtual memory. Virtual memory writes to memory only when the process requires it. Overall performance should improve as a result of this scalable memory management process.>
5. **Distributed Systems and Networks**: <A distributed system is made up of multiple software components that reside on different computers but operate as a single system. A distributed system, in my opinion, should be able to handle requests from various platforms. Implementing this Distributed System across multiple networks will provide a platform for all players to receive a high-quality product. >
6. **Security**: <Historically, Linux operating systems battled with security, but this has been addressed to an amazing degree over time. Authentication is the main security system. To access the parts of the system for which they are authorized, the user must enter a username and password during this process.>