

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.0

## 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)

[**Evaluation 4**](#_Toc115077325)

[**Recommendations 5**](#_Toc115077326)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 11/11/23 | Justin Angle | Initial Software Design |
| 1.1 | 11/24/23 | Justin Angle | Adding Evaluation |
| 1.2 | 12/5/23 | Justin Angle | Adding Recommendations |

**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)

Creative Technology Solutions (CTS) has a client, The Gaming Room, who has a game called “Draw It or Lose It”. At the moment, it’s only available on Android, and The Gaming Room would like to develop it to a web-based game to serve more platforms and customers. To solve this problem, we will create the application in Java utilizing object-oriented programming principles. This design will leverage the entity class, providing a common base for all entities with attributes.

## Requirements

*<* Please note: While this section is not being assessed, it will support your outline of the design constraints below. *In your summary, identify each of the client’s business and technical requirements in a clear and concise manner.>*

## [Design Constraints](#_2et92p0)

There are some design constraints to be cognizant of. The first one is the web-based environment, creating this game in this environment means that the game should be accessible over the internet and support multiple users. It would require ensuring the game can handle any network latency and cyber security considerations that may arise. Another design constraint would be the use of Java programming language. Utilizing Java the game is limited to the java-based frameworks and it means that we are to adhere to java coding and best practices.

Another design constraint would be the object-oriented design, this requires that we design classes with the proper encapsulation and inheritance. We also need to make it a flexible and extensible codebase by applying design patterns and abstraction techniques. We need to ensure the game can handle scalability and performance. This game should be scalable to meet the increasing user base it may potentially see. The user interface also needs to be user-friendly and intuitive so the user is able to navigate it easily.

## [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 below UML diagram shows the domain model of the game. It has several classes including Team, Player, GameService, and Game. The Entity class is the base class for the entities in the game. The GameService class is the central service the management of the games. This class maintains a list of games. It also provides methods to add games, retrieve games, and retrieve the game count. This class also follows the Singlton design pattern, which ensures only a single instance is accessible in the program. The Game class is the game in the program. Maintaining a list of the teams that are participating in the game and it provides methods to add teams and retrieve a game. The Team class is a team within the game and it maintains a list of players. The class provides methods to add players and the ability to retrieve a team using a string. The Player class is an individual player and provides the ability to retrieve the play by a string. These classes contain the principle of composition, where the Game class consists of Teams, and each Team consists of Players. This allows for a hierarchical structure giving the ability to manage games, teams, and players efficiently.

**"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** | The Mac has a flexible terminal which allows commands that can configure, access, or make changes to the server. Some advantages are that it is upgradeable and has multiple options for web hosting requirements. Some disadvantages are that many times Mac is proprietary and has a hard time working with other operating systems. | Similar to the Mac however it is cheaper to get. Pretty secured and good for web hosting. Some advantages are that the security flaws are usually caught before an issue arises. Some disadvantages are that it might be more difficult to find support for the applications for web hosting. | There is more software available to Windows than other operating systems. Majority of people use the windows OS. Some advantages are that is has good comfortability, less loading times, and meets resource requirements. Some disadvantages are that it’s susceptible to viruses, more than the other operating systems. | It is possible to implement web servers for mobile devices. Oracle is a company that offers mobile server implementation. The advantages of using Oracle is iOS development tools, android development tools, and the synchronization to Oracle. |
| **Client Side** | Mac clients would benefit from the familiarity and user friendly UI. However, it would require expertise and have a similar cost to windows as it requires OS purchase. | For Linux it would require a lot of expertise as it’s not as well known but low cost as Linux is not proprietary and does not require purchase of the OS. | Windows would have a low expertise and time required as it is well known. However, the cost would be similar to Mac as it is also proprietary and requires a purchase for the OS. | Mobile devices would provide flexibility to the clients, but require expertise and time. The cost would be a bit higher as the server would need to be in a centralized location. |
| **Development Tools** | Swift is a popular development IDE for Mac. Mac can run almost all languages including HTML, CSS, Java. And support libraries in the frontend like Python and Ruby. | Linux can utilize visual studio and eclipse. It can also work with many languages including the same that Mac can handle like HTML, CSS, and Java. It is also able to support the frontend and other languages like Python and Ruby. | Windows can run almost any program out there like eclipse, pycharm, and visual studio. There are many options as many IDEs are made for Windows. It can also support the same languages as Mac and Linux. | Using android and swift, you are able to make multiple applications. However to program these applications in the programing languages usually need a computer with an OS to run them. It would use many of the same languages as Mac, Linux, and Windows. |

## 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**: When considering all the operating platforms, I would recommend windows to be able to expand Draw it or Lose it to other computing environments. I believe windows is the best platform because it’s able to integrate with Android and also has cross platform solutions to be able to setup a development environment. Windows also is well known by a lot of people as it is the main operating system in production today. There is also a lot of emulators that would allow testing on almost all platforms.
2. **Operating Systems Architectures**: There are two modes on windows which are user mode and kernel mode. Kernel mode is more in depth and low level, it deals with outputs and inputs, memory management, networking, routines, and hardware management. While user mode is user facing and is what most users will interact with. Windows has a directory structure to be able to hold data. It also has support for multiprocessing and many hardware which allows for modularity in the system.
3. **Storage Management**: For storage management, there are many options as most everything works with windows, however I recommend Microsoft Azure as they have good prices and support and have continuous updates for their product. Azure has docker containers that leverage cloud storage instances. Cloud storage is effective with it’s cost efficiency and scalability and it’s ability to reach multiple people.
4. **Memory Management**: Windows 11 has improved its memory management over its predecessors which allows for faster load times from memory. Windows 11 is also able to take full advantage of DDR5 RAM which allows for a huge increase over DDR4. Every process can also utilize the entire virtual memory space, which would be more than enough for Draw it or Lose it.
5. **Distributed Systems and Networks**: This is another reason why I recommend Microsoft Azure as it has ease of access when we are dealing with distributed systems and networks. Azure has maximum uptime with cloud alerts, the Azure app, and monitoring solutions it has. This would be essential when dealing with multiple concurrent games in process. It also offsets the network load onto Azure so the Gaming Room can focus on the game and other features.
6. **Security**: If using Azure, it simplifies the logistics of the security of user’s personal data and information. Devices would use the Azure active directory to log on to the game. Azure also has extra security features like IP configuration for whitelist specific access. The option for VPN storage in the cloud. Databases can also have a IP whitelist, a password, and require SSL to be able to protect user data.