

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 3.0

## 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 | 11/12/22 | Russell Wheeler | This revision made additions to the following areas:  Executive summary, design constraints, domain model, title page, development requirements chart, and recommendations. |
| 2.0 | 11/26/22 | Russell Wheeler | This revision added to the developmental requirements chart regarding the implementation costs implementation. |
| 3.0 | 12/9/22 | Russell Wheeler | This revision added to the requirements chart regarding security. |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room needs a web-based game that can run on several platforms. A game titled “Draw It or Lose It”, the game is currently available on the Android Play store. The game will be team-based and will be played at round-to-round intervals, and each round will last one minute. During every round, the game will render images from a library and team members will need to guess the image before time runs out. If the team fails to give a correct answer the opposing team will be able to give a single guess at the image.

## [Design Constraints](#_2et92p0)

The Gaming Room wants to have this game available on all platforms including Android, IOS, Windows, and Mac. The game is already running on Android devices and the code will need to be rewritten or repurposed in order to function properly on all platforms.

Proposed requirements are:

* Each team will have multiple players assigned to it.
* A game will have the ability to have one or more teams involved.
* Unique game and team names will be used in the application
* Only one instance of the game can exist in memory at any given time.
* Application will run on multiple platforms

## [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 will create the relationships between the Game, Team, and Player classes. Each class will inherit information from the Entity class. The included UML showcases common references between classes like name and id. The Entity class is a superclass that will handle the relationships between each class. Each team will have a player or player, while each game will have a team or teams. The class GameService has a reference to Games, and Games has a reference to Team, and Team has a reference to Player.

**"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** | Flexible command input for use on Terminal to configure server, client access, and changes  OS X Server is priced at $499 USD for the 10-client version or $999 for the unlimited client version.  The hardware required is also listed at a starting price of $2,999 USD | A cost-effective version of Mac. Adaptable command input for use on Terminal  There are many server providers that allow someone to rent a virtual machine. This service typically starts at $250 USD and tops off at $1300 USD. | The most found as well as the most used operating system.    Windows server licenses range from $6,155 for the datacenter edition to $501 for the essentials edition of the sever packages | A server being run from a mobile device is prone to crashes and several other failures.  Costs for a mobile server are unknown as the whole process would need to be created from scratch. |
| **Client Side** | User friendly interface with a slightly higher learning curve than windows. Higher costs for end users that can be unaffordable to some.  Accounts for a small portion of the total users in the world.  Swift is a lesser-known coding language and will have to be considered for development. | Smallest consumer base with a very customizable interface. Extremely Large learning curve while being the most affordable.  The smallest user base of the listed categories. Little value for implementing multi-user support. | Largest consumer base that Is now competing with mobile devices. It has a familiar user interface as well as being affordable with a minimal learning curve.  The largest consumer base competes with mobile devices and is a must for multi-user support. | Competing with windows for the largest user base while being versatile, customizable, and offering flexibility. Development may be tough; however, it has made significant improvements to IDE compatibility.  Competing for the largest consumer base with windows implementing multi-user support here will reap large rewards. |
| **Development Tools** | Swift and X-Code are run-of-the-mill for Mac. While other languages are available these are designed for Mac. Several tools like Notepad++ can work on multiple devices.  XCode is a common IDE used for Mac development and costs $99 USD per year, per developer. | Support for almost every language and development environment on the market.  Python IDEs are often free, e.g., Notepad++. PyCharm is another popular Python IDE.    C/C++ IDEs are numerous – but not all are available for Linux. | Supports most languages and Development environments.  Microsoft’s Visual Studio is a vastly popular IDE and brings many plugins and integration options to the table.  Visual Studio costs from $45 to $250 USD per user, per year, based on features required. | Mobile app development has exploded recently, and now supports many languages for mobile development.  Most of iOS’s languages are exclusively developed in XCode.    XCode is listed as $99 USD per year per developer. |

## 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**: Windows is recommended for development due to its widespread popularity and support. There are several development environments to use with resources available for additional support.
2. The majority of servers are run using Linux and this is the recommended solution. This will not only save the company money in license costs but will not limit data center access as other operating systems and companies do. Linux offers good security and operability. It is the most common server platform and there are many tools available which include security software.
3. **Operating Systems Architectures**: Windows is recommended for development due to its widespread popularity and support. Subsequently, it will support the user with a unique and user-friendly interface as well as access to several graphical interfaces, multimedia, and web services.
4. **Storage Management**: Many machines are equipped with proper storage or could be upgraded to have adequate storage. Users are given the power to manage storage as they see fit. Installation wizards will allow a user to select the place of installation on their device. After installation, users can move applications and files to places of their choosing.
5. **Memory Management**: The development and production of this game will need to utilize a database or library of images. The images should not be stored in a location separate from the game on the machine and are recommended to store in the game files.
6. **Distributed Systems and Networks**: Separate operating systems need different development strategies and may require the development process to take place on different IDEs and languages. Several 3rd party development tools can be used to ensure the game can be run on different operating systems and platforms. I recommended that the servers could handle the high-density traffic that is coupled with server-based games. Servers should be robust and have proper security measures to allow such traffic.
7. **Security**: Most operating system platforms have built-in security for viruses and malware. The use of additional security services and programs can be used at the user's discretion. However, additional security measures are recommended for the server side of the game. Proper security measures will ensure a safe and pleasant environment. The use of automatic updates is recommended as new threats are found often.