

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.2

## 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 | 05/21/2022 | Nick Markel | Added summary, design constraints, Domain Model, evaluation, recommendation |
| 1.1 | 06/01/2022 | Nick Markel | Added development requirements table |
| 1.2 | 06/16/2022 | Nick Markel | System architecture recommendation |

**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)

Draw it or Lose it wants to create a web-based game that will render images from a library of stock drawings and a team guesses the puzzle before time expires. The team will get 30 seconds to guess before the remaining teams have 15 seconds to guess the puzzle. There are 4 rounds of play lasting one minute each.

## [Design Constraints](#_2et92p0)

* As this is a web-based game, it must be written in a programming language suitable for a web-based environment.
* Must have multiple teams with multiple players assigned to each team. Including customizable team names.
* Only one instance of the game is allowed in the memory at a given time.
* Needs a timer that ends and blocks users from entering input after 30 seconds and 15 seconds.

## [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)

OOP is used by encapsulating private variables and using setters and getters inside every class. Inheritance is incorporated with each class. GameService also feeds into the Game class. Game, Team, and Player class all feed into the Entity class too. Polymorphism is displayed as the classes lower in the UML act as parent classes to those higher in the classes. This UML shows the hierarchy of the program and what variables are allowed to be used where. Since all classes are connected except for GameService, all classes will be able to utilize the functions within each class.

**"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** | Utilizes Apache web server. Apache is behind 46% of all websites. Cost: starting at $60/m. | Also utilizes Apache for majority of services. It is very large and well known with a large database of instructions cost: $60/m. | Uses IIS mostly. Second to Apache web services. Cost: starting $501. Top end $6,155 for thousands of users. | Can be used for very small web services. Only usable by 1-5 people. Only usable on Android. Cost: Free |
| **Client Side** | Mac machines are expensive for initial purchase. They are a lot different from a windows machine and require some time to learn ins and outs from programmers. Keeping program vaguer will help with cross-platform. | Linux is free for build license but requires advanced knowledge for programmers. Linux build is not as secure and requires great knowledge of security. Linux is the best build to transfer to Mac and transfers well to windows. | Expensive license cost compared to Linux. Requires minimum expertise and time to learn. Comes with many different programs that promote cross-platform. | Extremely hard to create programs. Very convenient for on-the-go operations and reaching a larger number of people. Requires a lot of skill and time to ensure cross-compatibility. |
| **Development Tools** | Swift is main programming software. Python and perl are other good choices. VS code is highly recommended IDE along with pychram. | Java is a very popular Linux IDE thus eclipse is recommended. VS code and sublime can also be used and are respectable. | Large range of IDE’s can be used in windows. VS code again is most recommended and native to windows. JS, python, HTML all can be used in project. | Not many native IDE’s included on mobile devices. Will be limited on languages depending on app found. |

## 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**: The operating platform that is recommended for this application would be a Linux based system.
2. **Operating Systems Architectures**: Linux is the recommended operating platform for its application due to the features that Linux can provide that the other systems cannot. Starting off, Linux offers live updates to the server that would ensure more uptime for the game as it would not need to be taken offline for updates. Secondly, Linux is free to use, this means a lower project cost and the ability for that money to be used somewhere else. These two reasons are among many why Linux is the preferred operating platform for this project.
3. **Storage Management**: A Linux platform will help with storage management as Linux is a slimmer and smaller build than a windows build. Linux is more of a barebones system where you add the features you want to suit your application. On the other hand, a system like windows or mac comes pre-loaded with many features that you can choose from. Though, those other features are still present and will take up storage.
4. **Memory Management**: Linux offers both virtual and physical memory management on their server builds. This will allow the programmer to use either form of memory management to suit the application.
5. **Distributed Systems and Networks**: Just because Linux is used on the server-side of the program, does not mean that all users must use a Linux machine to access it. Any type of machine will be able to access the application as the server build is separate from the user end. A big upside to the network aspect of the Linux build is the ability to push updates or patches without bringing down the entire application. This will result in more game time for the user and no need for people to restart their applications when an update is pushed.
6. **Security**: Security is greatly better on the Linux system as the only people having access to the program are the creator and selected users. This ensures that there are no unauthorized persons entering the program source code. Also, Linux has a large open-source community that responds to known virus attacks as soon as they happen to ensure future security.