

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 | 1/18/2022 | Nicholas Truong | Filled in information |
| 1.1 | 2/10/2022 | Nicholas Truong | Edited previous information on Operating Platforms |
| 1.2 | 2/17/2022 | Nicholas Truong | Added 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)

The client, The Gaming Room, wants to develop a web-based game that serves multiple platforms based on their current game, Draw It or Lose It. The game has teams guessing the puzzle that is rendered from images from a large library of stock drawings as clues. There are four rounds that last one minute each and drawings are rendered at a steady rate and complete at the thirty second mark. If a team does not solve the puzzle, the remaining teams have an opportunity to solve the puzzle at the fifteen second time limit.

## [Design Constraints](#_2et92p0)

There needs to be one or more team. Each team has more than one player. Game and team names need to be unique. Only one instance of a game can be running at a time. It must be available 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 is a base class that creates a relationship between the Game, Team, and Player classes. Those three classes inherit from the Entity class. Each class will share common variables like “name” and “id” variables. The ProgramDriver class Is used to run the main() method, everything else is abstracted since you do not need to know how it runs in the background. The GameService class has private variables for encapsulation. The toString() method is an overload which is an example of polymorphism.

**"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)

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Mac has a flexible terminal command that lets you configure a server easier. It is very popular when it comes to web hosting. It is upgradeable and offers different web hosting options. It is more expensive compared to the other OS. Mac has many open source libraries available also. | Linux is secured and most clients prefer Linux. Linux offers stability and is open source. Linux is used less on desktop and more on other devices such as smartphones, cars, and home appliances. Linux also leads in cloud infrastructure. Any security flaws are usually caught before any issue arises. It can be difficult to troubleshoot. | Windows has more software compared to the other OS. Many people use Windows and it is considered the dominant platform. It has high resource requirements, low loading time, and is user friendly. It is more susceptible to viruses may have security flaws as well as poor tech support. | Mobile devices would not make for good servers due to lower specifications and poorer security. The advantage to using a mobile device would be that almost everyone has one and that it can be lower cost. |
| **Client Side** | It costs $19.99 to add a macOS server from the Mac App Store.  It should take a moderate amount of time and expertise to implement. | It costs somewhere between $0.99 to $3.95 to host a server with Linux.  It should take a large amount of time and expertise to implement due to a large learning curve. | It costs around $2 to $20 to host a server with Windows, but buying the license also adds to the cost. It should take a small amount of time and expertise to implement. | Cost is moderate. It should take a small amount of time and moderate expertise to implement. |
| **Development Tools** | Mac has Xcode for the popular IDE but it can use other IDEs like Eclipse, Atom, and VS Code. Mac has a popular programming language called Swift. | Linux can use IDEs such as Eclipse, Atom, Sublime Text. Linux primarily uses C as a programming language. | Windows can use IDEs such as Eclipse, Atom, Sublime Text. Windows is very flexible when it comes to programming languages, but it primarily uses C, C# and C++. | Mobile devices have IDEs. AIDE for Android. Android devices use C, C++, and Java. iOS uses Swift primarily for its programming language. |

## Recommendations

1. **Operating Platform**: I recommend starting on Windows platform as it has minimum entry for apps.
2. **Operating Systems Architectures**: There is the hardware, kernel, shell, and application. Hardware is anything physical like the CPU. Kernel is the bridge between applications and actual data processing done by the hardware. The shell is the software that provides the interface for users and access to the services of the kernel.
3. **Storage Management**: Windows 10 has storage management built into it. It is called Storage Sense and it can be used to manage files on a hard drive. Storage sense detects when a hard drive is reaching capacity and offers tools and maintenance to optimize storage space.
4. **Memory Management**: Windows manages its own memory depending on if it is the 32- or 64-bit version. For the 32-bit version, each process has its own virtual address space that enables addressing up to 4 gigabytes of memory. For 64-bit, each process has a virtual address space of 8 terabytes.
5. **Distributed Systems and Networks**: Consider publishing the game through a game engine such as Unity. Unity is capable of publishing to multiple platforms. By maintaining the server and making sure there are backups such as Cloud saving, outages or connectivity problems will not be as bad.
6. **Security**: By using the REST API, you can separate client and server. REST API would work with any operating platform and keep information such as passwords and usernames private. Windows has built in security. In the past, it was called Microsoft Security Essentials and you can get extra software such as Norton, Kaspersky, and Malwarebytes for extra protection.