

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 | 11/19/2022 | Katrina Brodski | Added information in Executive Summary, Design Constraints, Domain Model, Evaluation, and Recommendations. |
| 1.1 | 11/27/2022 | Katrina Brodski | Updated Evaluation table. |
| 1.2 | 12/11/2022 | Katrina Brodski | Updated Recommendations section. |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room would like to develop a web-based game that can run on multiple platforms but is only available on Android. Their game is called “Draw It or Lose It”. Multiple teams with multiple players per team will play four rounds of play lasting one minute each. If the team does not guess the puzzle before time expires, the remaining teams have an opportunity to offer one guess each to solve the puzzle with a 15-second limit.

## [Design Constraints](#_2et92p0)

The game must have the ability to have one or more teams involved and each team will have multiple players assigned to it. Game and team names must be unique to allow users to check whether a name is in use when choosing a team name. By creating unique identifiers for each instance of a game, team, or player; there shall only be one instance of the game to exist in memory at any given time.

## [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)

Entity creates a relationship between Game, Team, and Player class. These three classes all inherit and receive information from Entity. Example: Game id, Team id, and Player id. Game service grabs any information needed from Game, Team and Player classes as needed for the code to run properly and so the players receive the correct output. Example such as getGameId, getPlayerId, and getTeamId.

**"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** | Server has various options for different web hosting requirements. Apple server isn’t a common web hosting service. | Server is open source (Free) which results in being cost effective, secure but with less support. | Server is paid but more secure and more support but more expensive. | Better compatibility and cost-effective. Highly selective to mobile devices with poor security. |
| **Client Side** | Average expertise and time is required with cost similar to Windows. Must ensure the application works across all devices. | Extreme expertise and time is required but minimum cost. Must ensure the application works across all devices. | Less time required and cost is similar to Mac. Must ensure the application works across all devices. | Capable of seeing updates in real time at any time and place. Harder to implement vs other devices. |
| **Development Tools** | XCode uses a programming language called Swift ( C ) | Eclipse using C for the default programming language. | Visual Studios | Use both Android and Swift to create. Both languages can run on all systems within this table. |

## 

## 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**: Based on the information within the Developmental table, I would recommend using the Windows operating system due to having a wider variety of IDE’s, smaller expertise is needed and more cost effective with the use of Visual Studio. Such languages that can be used are C++ and Java considering both languages are object oriented and can provide security within the code.
2. **Operating Systems Architectures**: There are multiple rings/layers to Windows architecture; the hardware, kernel, shell, and application. The hardware is the physical means of a computer where the memory, storage and other devices are held. The kernel portion is the most trusted part to the operating system. The kernel controls all the important functions of the hardware. This is the case for Linux, Apple, and Windows as well as smartphones. The shell provides an interface for users to an operating system which provides access to the services of a kernel. This has capabilities such as viewing the contents of directories.
3. **Storage Management**: Using the automation option within storage management, a tool can be used to automatically update a shared database whenever a change is made on a local computer rather than manually updating. This will make sure that the database contains updated information for all users and prevents users from viewing outdated information if someone forgets to submit changes.
4. **Memory Management**: This game will require either a database or a library of many pictures. The memory allocation allows for easy storage of pictures outside of the default picture folder which allows the project to be kept together in a secure way. The memory for the photos being used will be 4GB alone, so using a 64-bit configuration, this will give plenty of room so the game will run smoothly on all operating systems.
5. **Distributed Systems and Networks**: Microsoft uses CIF’s (common Internet file system) which is used in conjunction with user authentication (user name and password) to create a network login which decided whether to allow or deny access to a requested file system. Using this, admin users would have access to change or modify files and game users have access to use files for the purpose of the game.
6. **Security**: Windows comes with built-in security on every Windows device which makes the security high. Done in the background of the computer running, it scans for malware, viruses and any security threats. By restricting access to users to particular only elements within the game, this provides security for the company. Using a client-server model, only the client can make requests to the server and the server will provide the information requested and nothing else. The best ways to protect user information is having a secure code, using multi-factor authentication and encrypting the application and data. Using an object oriented programming language provides security in the code by making classes private so the public cannot view it but anyone within the company can. Using multi-factor authentication provides higher security for users such as device ID, client certificates and OTP (one-time password) can help reduce the risk of unauthorized access.