

Draw it or Lose it!

# **CS 230 Project Software Design Template**

Version 1.0

## 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 | 15 May 2022 | Sergio Mateos | The documentation of Creative Technology Solutions will provide an idea of how the game would be develop, specification and trouble that the time might face. |
| 2.0 | 31 May 2022 | Sergio Mateos | Evaluate characteristics, advantages, and weakness of the operating platforms |
| 3.0 | 12 June 2022 | Sergio Mateos | Analyze the characteristics of a techniques specific to various system architectures |

## [Executive Summary](#_sbfa50wo7nsh)

The Game Room wants to develop a web-based game that will serve multiple platforms on their current game, the game is currently available on Android only. The “Draw It or Lose It” it’s a team game in which the opponents of rival teams must guess what image has been pulled from the library until the time runs out. The team will have four rounds of one minute each, and if not answered, the opposite team will have 15 seconds to try to solve the puzzle.

## [Design Constraints](#_2et92p0)

1. The game will allow access to multiple teams.
2. Each team will be integrated with multiple team members.
3. Game and team names must be unique to verify whether the name is being used.
4. The instance of the game can exist in memory at any time which will be accomplished by creating a unique identifier for each instance of a game, team, or player.

## [System Architecture View](#_ilbxbyevv6b6)

The Game Room specifies these requirements to be met while Creative Technology Solutions write the code and develop the app and website. The developing team has the Android platform, since The Game Room must be on multiple platforms, the development team needs to make sure the game can be playable on iPhone, Mac, and Linux (for the computer website) as well.

## [Domain Model](#_8h2ehzxfam4o)

The Gaming Room UML makes **Entity** a superclass which means that subclasses like **Game**, **Team**, and **Player** will inherit the attributes like id and name, and the methods like getId, getName, to String, etc. The subclasses will have their unique methods and attributes. **Game**, **Team**, and **Player** have an association, **Game** has an association with **Team, Team** has an association with **Player,** and **Game** has a unique association with **GameService.** All these subclasses have a multiplicity association from 0 to many. **SingletonTester** has an association with **ProgramDriver**, this association is unique since **SingletonTester** uses **ProgramDriver.**

**"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** | Configuration and change are adaptive on the servers but with a high cost of server use. GUI is easy to understand and use. The security is high/moderate which helps to secure the application and the users. | Linux has a problem finding apps to support the web hosting. The navigation is more difficult requiring a high level of expertise.  Very secure and excellent server implementation. The inexpensive scaling tools make change reachable. | Larger software available and community. Windows (Azure) have a high price. GUI is easier to understand and provides plenty of options for developers.  Windows tend to be less secure. | It is easier to reach a larger community, in this case, there are more Android users than IOS worldwide. The problem would be implementing the game across all the different platforms.  The security would be poor. |
| **Client Side** | Cost: High cost.  Time: Moderate  Expertise: Moderate knowledge. | Cost: Low cost.  Time: High.  Expertise: High knowledge. | Cost: Moderate cost.  Time: Low.  Expertise: Low/Moderate Knowledge. | IOS and Android  Cost:  Time: Both can be updated at any time.  Expertise: Usually IOS is more difficult to implement compared to Android since IOS is not set a stone. |
| **Development Tools** | Programming Language: Swift and Objective C  Development tool/IDE: Visual Studio (Free and Open-Source), XCode (Free), CLion (Free and Paid), AppCode (Free and Paid), PhpStorm (Paid), Atom (Paid), Brackets (Free and Open-Source), Sublime Text (Free and Paid), Espresso (Paid), TextMate (Free and Paid). | Programming Languages:  C, C++, and Bash.  Development tool/IDE: CLion (Free and Paid), Eclipse (Free and Open-Source), Code::Blocks (Free and Open-Source), Visual Studio (Free and Open-Source), Brackets (Free and Open-Source), EMACS (Free), Visual Studio (Free and Open-Source). | Programming Languages:  Visual Basic, C#, and JavaScript.  Development tool/IDE: Visual Studio (Free and Open-Source), Eclipse (Free and Open-Source), Atom (Paid), Azure (Free and Paid), Cloud 9 (Free and Paid), Axure (Paid). | Programming Languages:  IOS: Swift.  Android: Kotlin and Java.  Development tool/IDE:  IOS: XCode (Free)  Android: Android Studio (Free) |

## Recommendations

1. **Operating Platform**: The Gamming Room should use Windows as an Operating Platform for the app “Draw it or Lose it”. Windows do not require a higher level of expertise and the operating system is designed to run on server hardware. Also, Windows server operating system will not run into a shortage when working with different IDEs.
2. **Operating Systems Architectures**: Windows servers operating system has a large amount of array and lists, but Widows provide the capability to manage files in the memory as conveniently as possible. Windows provide the ability the developers to access to work with many different programming languages.
3. **Storage Management**: Widows can allow greater storage management by storing the app directly to the phone (in the main memory) or hard drive. If the user decides to store the game on the phone this will allow for a faster process when the gaming is loading. Also, the manipulation of files will let the developer create an infrastructure that allows them to find the files quicker.
4. **Memory Management**: Windows server operating system allows memory management options including virtual or physical. Also, the memory allocation facilitates the storage of pictures in different files, which allows to keep the whole project together and secure.
5. **Distributed Systems and Networks**: Windows server operating system as the distributed systems and network will create simple communication between processors, and implement we need an IDEs that would be supported by any device. So, once it’s been supported and implemented game can be exported and imported to different games file on Mac, Linux, and Mobile devices. Verification of servers to prevent any crash so the app would support many players.
6. **Security**: Windows has built-in security protection that protects the user from data going in and out of the system. This system can scan malware, viruses, and security threats. Since Windows had security issues, it’s recommended to use extra protection as a VPN service. The VPN will protect the client’s data and accounts from hacking and other criminal activities. In this case. Mac has higher security which is been proven that is more complicated to break compared to Windows.