

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 | 09/17/2022 | Greg Monti | Updated information |

**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 company ‘Gaming Room’ is looking to develop a new game for their platform, known as “Draw It or Lose It.” Gaming Room would like this to be a web-based game that can perform on multiple platforms and OS. It is currently only available on the Android platform and are looking to expand it to other platforms as well. The purpose of this game is multiple teams competing to guess what image is shown on screen until the timer runs out. If the team fails to guess the image, the opposing team will get a turn each to guess.

## [Design Constraints](#_2et92p0)

* Involving one or more teams
* Each team consisting of multiple individual users
* A list must be checked of existing team and player names, so there are no duplicates
* Must run on multiple platforms
* Must run on various machines and OS

## [Domain Model](#_8h2ehzxfam4o)

The Entity class has a relationship between Game, Team, and Player classes. With Entity being the superclass, these three classes inherit its traits from the Entity class. We then look at GameService class. We can read this as GameService “has a” Game, Game “has a” Team, and Team “has a” Player. Each instances of these “has a” inherits traits from the previous 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** | -not great web hosting services  -great virus protection | -most secure OS  -can catch problems before they happen  -great virus protection | -more applications  -the more used OS  -virus protection not great | -pages take to long to load  -more easily accessible  -weakness of running game through web browser |
| **Client Side** | -more expensive  -more expertise required  -compatibility of code? | - the most expertise  -lower cost  -compatibility of code? | -least expertise  -slightly cheaper than Mac  -compatibility of code? | -flexibility of being anywhere  -will be more difficult than other OS |
| **Development Tools** | -using swift | -universal tools that can be used on other OS | -easier than linux, but can use the same tools like Ecplise and visual studio | -using swift again since we already have an Android version |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tool** | **Description** | **Linux** | **Mac** | **Windows** |
| Eclipse | Free to use, easily trainable, open source, beginner friendly | Yes | Yes | Yes |
| Visual Studio | Free to use, easily trainable, open source, beginner friendly, Microsoft based | No | No | Yes |
| Axure | Team oriented, multiple people can work on project at the same time, Microsoft based | No | No | Yes |
| Azure | Web management, cloud saving and apps, Microsoft Based | No | No | Yes |
| Atom | Free, open source, flexible with coding, multiple projects in one viewer | Yes | Yes | Yes |

## 

## 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**: I would recommend that we go with Windows OS. This is because we can reach the majority of people and it is the easiest to implement of the others. It will take the least amount of cost and expertise as well.
2. **Operating Systems Architectures**: Windows has the most customization and accessibility compared to the other platforms. It also provides service for all windows based applications, which is a lot of them.
3. **Storage Management**: In Windows OS, it comes with a nifty built in tool to clean up unused and unwanted files to make more room. There is also options for cloud saving data and applications.
4. **Memory Management**: Memory management will be important as we will have a large file with thousands of pictures. We will create a database to store all images to be able to access them more easily and secure.
5. **Distributed Systems and Networks**: Each system and OS are different, and we will need a way to translate from Windows OS to the others easily. We must find an application that can run on any device, and use the code within that domain, so that we can easily port it to the various systems. Need to make sure the network is secure and large enough to hold a large player base.
6. **Security**: Although windows does come with a built-in firewall, it is not the best in the business. We may need to look to implement a third party firewall to make files more secure for the server as well as the client side.