

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template** 1](#_Toc115077317)

[**Table of Contents 2**](#_Toc115077318)

[**Document Revision History 2**](#_Toc115077319)

[**Executive Summary 3**](#_Toc115077320)

[**Requirements 3**](#_Toc115077321)

[**Design Constraints 3**](#_Toc115077322)

[**System Architecture View 3**](#_Toc115077323)

[**Domain Model 3**](#_Toc115077324)

[**Evaluation 4**](#_Toc115077325)

[**Recommendations 5**](#_Toc115077326)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 11/30/23 | Cynthia Craig | Added Summary and Design Constraints |
| 1.1 | 12/1/23 | Cynthia Craig | Added Domain Model, Evaluation, and Recommendations |

## [Executive Summary](#_sbfa50wo7nsh)

The Android app game “Draw It or Lose It” will be replicated in a web browser on multiple platforms. An image will be generated slowly over the timespan of 30 seconds and one team will try to guess the image as it is being drawn. If the team does not successfully guess the image, the other teams have 15 seconds to put in a single guess as to what the image is.

## [Design Constraints](#_2et92p0)

* Compatibility across different operating systems and browsers. It should be able to be accessed through several different methods for all players.
* The game depends on low latency because it relies on real time rendering of an image to all involved parties.
* Game and Team names must be unique.
* Only one instance of the game can be running at a time.

## [Domain Model](#_8h2ehzxfam4o)

The ProgramDriver class contains the application’s main method and performs a singleton test using a SingletonTester instance. The Entity class is the common ancestor of the Game, Team, and Player classes, which all inherit from it. The GameService class has a zero to many connection with the Game Class. Games may have zero to many Teams, and Teams may have zero to many Players.

**"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** | MacOS has easy accessibility and configuration. It is easy to design and to access to make changes and has an easy to use graphical interface. | Most affordable. Linux is easy to configure and accessible, but it is difficult to navigate the platform. | Windows is expensive, but has more widely available software compared to other OS. It also has very user friendly GUI. | Mobile device specifications vary. It’s not logical to host. |
| **Client Side** | Mac is an expensive option for users that requires a moderate about of skill and time to use. Decent skill is needed to navigate the OS. | Linux is the most affordable option for users, but requires the most skill and time to use. | Windows is more expensive than Linux systems, but not more than Mac. Users do not need a lot of skill or time to navigate the Windows OS. | Applications are hard to use when created for other platforms. More difficult to implement than other devices. |
| **Development Tools** | Languages consist of, but are not limited to HTML, CSS, and JavaScript. Dev tools for Mac systems include PyCharm, Eclipse, and Visual Studio, as well as online development tools. | Languages consist of, but are not limited to HTML, CSS, and JavaScript. Dev tools for Linux include PyCharm, Eclipse, and Visual Studio. | Languages consist of, but are not limited to HTML, CSS, and JavaScript. Dev tools for Windows include PyCharm, Eclipse, and Visual Studio. | Languages consist of, but are not limited to HTML, CSS, and JavaScript. Dev tools include PyCharm, Eclipse, and Visual Studio. |

## 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 recommend the Gaming Room to use Windows for its servers.
2. **Operating Systems Architectures**: Although not as cheap as Linux, Windows is very easy to work with and for users to navigate. It also has much more widely available software than the other operating systems.
3. **Storage Management**: For storage, I would highly recommend using Microsoft Azure. They have very competitive prices and continue to receive updates and support. It is also cloud based storage, which is quickly becoming a widely used storage solution due to affordability and scalability.
4. **Memory Management**: Windows 11 is the newest version of Windows OS, and while it uses more RAM than the previous version (Windows 10), it provides increased functionality and features.
5. **Distributed Systems and Networks**: By utilizing Microsoft Azure as the cloud service provider, you would also be gaining ease of access concerning distributed systems and networks. It offers several different global traffic distribution solutions to suit the different needs of each company.
6. **Security**: Microsoft Azure also offers simplistic safety features that protect user information and personal data. It offers whitelist specific access to resources based on IP, protects the database with IP whitelist access, passwords, and SSL connectivity, and has an option for storage in a VPN.