

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 3.0

## 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 | 03/20/22 | Tim Painter | New Project |
| 2.0 | 4/2/22 | Tim Painter | Continuation of project |
| 3.0 | 4/17/22 | Tim Painter | Final completion of project |

## [Executive Summary](#_sbfa50wo7nsh)

An application needs to be built that mimics the television game “Win, Lose, or Draw”. The app will render images from a library of stock drawings as clues. Each game is four rounds of one minute each. Drawings will be rendered steadily and will be complete at the 30 second mark. If the team does not guess the puzzle before time expires, the remaining teams have one guess to solve the puzzle within a 15 second time limit.

This project will require at least one developer and one tester.

## [Design Constraints](#_2et92p0)

* Will have the ability to have one or more teams involved
* Each team will have multiple players
* Unique game and team names. App allows users to check if name is in use.
* Only one instance of game in memory at any given time.
* Create unique identifiers for each instance of game, team or player
* Must operate on multiple platforms (currently available as an Android app only)

## [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)

Game, Team, and Player class all inherit information that is gathered with the class Entity. GameService references from Game, Team, and Player. Each of these classes gather information for each of their namesakes. All of the classes inherit info from the other classes.

**"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** | Server easily configurable.  Works best with Mac clients.  Quite often used in web hosting but less preferred. | Less costly.  Preferred choice for web hosting services.  Stable and reliable.  Linux applications are more difficult to find for support of web hosting. | Most widely used OS.  Much more software available.  Less training required.  Can be susceptible to viruses. | Limited bandwidth for programming.  High portability.  Not as secure. |
| **Client Side** | Mac can be more costly than Windows. Especially if new client computers are required to convert to Mac.  May be a learning curve to convert Windows user to Mac. | Huge learning curve since much less used than either Mac or Windows.  Most software is open source. | As with server side, since widely used, less of a learning curve and less training required. | Most popular operating systems since most people always have them available close at hand.  Updates can be done anywhere with a connection.  Implementation can be difficult depending on user. |
| **Development Tools** | Python, Java, and Ruby can all be used for development. | Works with Eclipse IDE.  Many programming languages can be used with Linux. | Many IDE’s available to utilize on Windows.  Most programming languages can be used. | Both Android and Swift can be used on both Apple and Android devices.  HTML, CSS, Python, PHP, Ruby, Java and Javascript can all be utilized for programming. |

## 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, at least initially, to start with Windows devices. This would eliminate learning curves and lower costs, since Windows is currently being used by the company.
2. **Operating Systems Architectures**: Windows services that are utilized by Windows applications to show a GUI to make navigation simple for users. Also, many IDE’s can be used with the Windows platform.
3. **Storage Management**: Windows makes storage management easy through storage sense which requires only a simple configuration. Additionally, in the future, a cloud server can be added for unlimited expansion.
4. **Memory Management**: In Windows, both physical and virtual memory can be utilized. The advantage of virtual memory is that it is essentially an expansion of the existing physical memory.
5. **Distributed Systems and Networks**: It is recommended that an IDE be used that could be used across all necessary platforms. This makes it easy to distribute across all platforms and help with dependencies. Also, any servers being used need to be able to handle a large amount of users and prevent any possible outages or loss of connections.
6. **Security**: Although Windows does provide security within the platform, it is recommended that a more robust third-party security package be used. One that automatically updates and scans for viruses, malware, and other security threats. Also, user logins should be used that could limit what different users will have access to.