

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

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

[**Table of Contents 2**](#_30j0zll)

[**Document Revision History 2**](#_3znysh7)

[**Executive Summary 3**](#_2et92p0)

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

[**Design Constraints 3**](#_1t3h5sf)

[**System Architecture View 3**](#_4d34og8)

[**Domain Model 3**](#_2s8eyo1)

[**Evaluation 4**](#_17dp8vu)

[**Recommendations 5**](#_26in1rg)

## [Document Revision History](#_lnxbz9)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 11/15/24 | Frankie Lawrence | Initial documenting of software design template. |

**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](#_35nkun2)

The purpose of this document is to outline the design of The Gaming Room’s Draw It or Lose It. This application draws loose inspiration from the 1980s television game Win, Lose or Draw, where teams compete to guess what is being drawn. In this application, images from a photo library will be rendered, and an active team will have the opportunity to score points by guessing the image correctly. If the active team fails to guess within their time allotment, the opposing team will have a brief opportunity to guess. After four rounds of play, the team with the most points will be declared the winner.

## Requirements

* **Puzzle/Photo Library Management**:
  + The game must access a large library of stock drawings to provide puzzle clues.
* **Platform Compatibility**:
  + The game must support **web-based functionality** and be accessible across **multiple platforms**, transitioning from the current Android-only app.
* **Round Timing and Gameplay**:
  + Each round must last **one minute**, with the drawing fully rendered by the **30-second mark**.
  + Teams have **15 seconds** to guess the puzzle if the primary team fails.
* **Environment Setup**:
  + The client requires guidance on setting up the environment for developing and running the application.

## [Design Constraints](#_1ksv4uv)

* **Business Constraints**
  + Platform Expansion - The application is moving from android-only to a web based solution. This will be a large step for the company to make.
  + User familiarity - Web based version must retain similar functionality and end user experience as the existing android version.
  + CTS is providing The Gaming Room (TGR) the initial environmental setup, as such, CTS must provide good documentation for future deployments and maintenance.
* **Techincal Constraints**
  + Parent company of Creative Technology Solutions (CTS), Southern New Hampshire University has asked all CTS applications to be designed in Java.
  + The web application requires an active internet connection. It will utilize web sockets to ensure multiple players remain connected to the game server and receive real-time updates concurrently.
  + The application will ideally require a database storage system to facilitate the efficient uploading, removal, and management of photos used as game hints.
  + The application must be compatible with multiple platforms, including web browsers on desktop and mobile devices, ensuring a seamless user experience regardless of the device.
  + As the application comes online, security systems should be in place to protect company and player information.
  + The application will utilize an iterator design pattern to parse through teams and players to ensure players and teams cannot have the same identity.
  + The application will utilize a singleton design pattern to ensure only one instance of a game can be active at a time.

## [System Architecture View](#_44sinio)

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](#_2jxsxqh)

This UML diagram outlines multiple key relationships and object oriented principles that will help organize the application’s code base.

The primary entry point to the application is the **GameService class.** The **GameService class** implements the **Singleton** design pattern ensuring that only one instance of **GameService** exists at runtime to manage game logic and data. It hosts multiple instances of the Game classand holds many methods for retrieving game and player data.

The next primary class of the application is the **Entity class**. This class demonstrates inheritance and polymorphism.Game, Team and Player classes all inherit directly from the **Entity class.** **Entity class** provides these child classes with unique identifiers and accessors. The unique identifiers in the **Entity class** will help our Itertator pattern parse through each of the following to ensure each child class instance is unique.

The **Game class** as mentioned previously is hosted by the GameService class and hosts multiple teams playing the game.

The **Team class** has the ability to add multiple players to an instance.

The **Player class** will be the class that acts directly with the game functionality once that portion of the application gets built out.

During development the **SingletonTester class** hosts the majority of our unit tests.

Lastly, the **ProgramDriver class** acts as our main() function for the program.

**"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](#_z337ya)

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** | <Evaluate Mac for its characteristics, advantages, and weaknesses for hosting a web-based software application.> | <Evaluate Linux for its characteristics, advantages, and weaknesses for hosting a web-based software application.> | <Evaluate Windows for its characteristics, advantages, and weaknesses for hosting a web-based software application.> | <Evaluate Mobile Devices for their characteristics, advantages, and weaknesses for hosting a web-based software application.> |
| **Client Side** | <Determine the software development considerations (cost, time, expertise) that are necessary for supporting multiple types of clients as they pertain to Mac.> | <Determine the software development considerations (cost, time, expertise) that are necessary for supporting multiple types of clients as they pertain to Linux.> | <Determine the software development considerations (cost, time, expertise) that are necessary for supporting multiple types of clients as they pertain to Windows.> | <Determine the software development considerations (cost, time, expertise) that are necessary for supporting multiple types of clients as they pertain to Mobile Devices.> |
| **Development Tools** | <Identify the relevant programming languages and tools (IDEs and other tools) that are used to build this type of software for deploying on Mac.> | <Identify the relevant programming languages and tools (IDEs and other tools) that are used to build this type of software for deploying on Linux.> | <Identify the relevant programming languages and tools (IDEs and other tools) that are used to build this type of software for deploying on Windows.> | <Identify the relevant programming languages and tools (IDEs and other tools) that are used to build this type of software for deploying on Mobile Devices.> |

## 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**: <Recommend an appropriate operating platform that will allow The Gaming Room to expand Draw It or Lose It to other computing environments.>
2. **Operating Systems Architectures**: <Describe the details of the chosen operating platform architectures.>
3. **Storage Management**: <Identify an appropriate storage management system to be used with the recommended operating platform.>
4. **Memory Management**: <Explain how the recommended operating platform uses memory management techniques for the Draw It or Lose It software.>
5. **Distributed Systems and Networks**: <Knowing that the client would like Draw It or Lose It to communicate between various platforms, explain how this may be accomplished with distributed software and the network that connects the devices. Consider the dependencies between the components within the distributed systems and networks (connectivity, outages, and so on).>
6. **Security**: <Security is a must-have for the client. Explain how to protect user information on and between various platforms. Consider the user protection and security capabilities of the recommended operating platform.>