

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 3.0

## 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 | 01/20/23 | Dennis Renfro | Completed Executive Summary, Design Constraints, and Domain Model sections. |
| 2.0 | 2/5/23 | Dennis Renfro | Evaluate Platforms |
| 3.0 | 2/18/23 | Dennis Renfro | Architecture Recommendations |

**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 Gaming Room wants to develop a web-based version of a television game from the 1980s called “Win, Lose, or Draw” that can run on multiple platforms. The game will be called “Draw It or Lose It”. The purpose of the game consists of four rounds of play lasting a minute each. The game will have the ability to have multiple teams and each team can have multiple players. When a drawing is selected from a library of images, one team guesses till time runs out. If not answered correctly, the remaining teams have an opportunity to guess within a 15 second time limit.

## Requirements

*<* Please note: While this section is not being assessed, it will support your outline of the design constraints below. *In your summary, identify each of the client’s business and technical requirements in a clear and concise manner.>*

## [Design Constraints](#_2et92p0)

* Must be written in a web-based language in order to be fully functional in any web-based environment.
* Must be written to allow multiple teams with multiple players.
* Game and Team names must be unique to allow users to check whether the name is in use when choosing a team name
* Only one instance of the game can exist in memory at any given time.
* Must run on multiple platforms

## [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 classes each inherit from Entity superclass allowing the attributes and operations contained in Entity to be written once and used by each of the classes inheriting from it. The GameService, Game, Team, and Player classes are all associated with each other. The ProgramDriver class drives the package and inherits/uses the SingletonTester class while doing so.

**"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** | Flexible terminal commands to configure the server, access, or make changes.  Characteristics It is popular in web hosting  Advantages It is upgradeable  Weaknesses: It is less preferred for web hosting services | Flexible terminal commands to configure the server, access, or make changes.  Characteristics Secured, most preferred.  Advantages Security flaws are caught before they become an issue, it is the most preferred choice for web hosting services  Weaknesses: It is more difficult to find applications to support the web hosting required needs. | More software available compared to other OS.  Characteristics It is dominant to the other platforms.  Advantages High resource requirements, less loading time, high comfortability  Weaknesses: easy virus susceptibility, poor tech support | It's better if the server is immobile and can be tracked in a single place. Specifications are better in other devices.  Characteristics More popular, high portability.  Advantages Have a wider reach, better compatibility, cost-effective  Weaknesses: It is highly selective to various smart mobile devices Poor security |
| **Client Side** | Medium expertise and time required. Medium cost. | Maximum expertise and time required. Minimum cost, not user friendly | Minimum expertise and time required. Medium cost and user friendly | Harder to implement. Highest cost. |
| **Development Tools** | Java, Python, PHP, and Ruby. | Java, Python, PHP, and Ruby. | Java, Python, PHP, and Ruby. | Java, C++, Python, PHP, and Ruby. |

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 windows platform for greater versatility including memory management, storage, security and many other issues that will be encountered.
2. **Operating Systems Architectures**: Windows provides services used by all Windows-based applications that enable applications to show a Graphical User Interface (GUI) while accessing system resources and much more. These applications also refer to Graphics and Multimedia, messaging, and web services.
3. **Storage Management**: The amount of storage is going to vary month to month and cloud storage is the way to meet this demand. Cost is reduced because the user will have access to the amount of storage space that is necessary and the client avoids the cost of unused storage space. Cloud storage can be used to provide flexibility and reduce data storage costs. Cloud storage makes it easier to access the files through the internet.
4. **Memory Management**: A database will have to be created that allows memory for 200 high definition image files, each one approximately 8 megabytes in size. Virtual memory and memory allocation should be employed. Virtual memory allows for the expansion of memory access in addition to the standard RAM storage available in the system. Memory allocation allows for easy storage of pictures outside of the default picture folder. This allows you to keep your whole project together in a more secure area on your computer. This includes when you’re working with your IDE and opening files from it to create the game.
5. **Distributed Systems and Networks**: A distributed system is a computing environment in which various components are spread across multiple computers (or other computing devices) on a network. The distributed system involving the game application, Draw It or Lose It, involves a multitude of networked computers that interact with a central server for data storage, processing and a common goal. The most common form would operate on the internet with cloud-based virtual instances being created and terminated as needed. Server and backup capabilities should be addressed because they determine the strength and durability of the system.
6. **Security**: Windows comes with built-in security protection software. An outside source should be used to provide the security and protection against malware that could compromise the system. Security is the most important part of the process with operating systems. Whatever preventive measures that are necessary to safeguard user information and maintain the privacy methods of the company should be employed. The client should have software in place to authenticate the user to determine if the person is who they say they are. User’s information should be encrypted as a safeguard. User authorization is best handled through role-based access control (RBAC) that promotes least privilege. Users should be assigned to player roles that allow them to play games and manage teams but make no app modification. Administrators can be assigned to roles granting additional access like managing library files, editing puzzles, or adjusting app configurations.