

Draw It or Lose It

# **CS 230 Project Software Design**

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 4**

[**Domain Model 4**](#_Toc115077324)

[**Evaluation 5**](#_Toc115077325)

[**Recommendations 7**](#_Toc115077326)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 03/19/2023 | James Lee | Updated the Cover page  Updated the Executive Summary section  Updated the Requirements section  Updated the Design Constraints section  Updated the Domain Model section |
| 2.0 | 04/02/2023 | James Lee | Updated the Evaluation section |
| 3.0 | 04/12/2023 | James Lee | Updated the Recommendations section |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room is a gaming company that owns a game application, Draw It or Lose It, that is available as an application for Android systems. In order to scale their business and increase their customer base, The Gaming Room plans on expanding their Draw It or Lose It game onto multiple operating platforms as a web-based game. The Gaming Room has retained the services of Creative Technology Solutions (CTS) to set up the distributed environment and develop the web-based version of the Draw It or Lose It game.

Draw It or Lose It is broadly similar to a game from a TV show from the 1980s called *Win, Lose or Draw*, where the contestant teams competed to guess the object or thing being drawn on an easel. For the Draw It or Lose It game, the application will render images selected from a large library of stock images as clues. These images will be partially rendered at a constant rate until the 30-second mark, when the images will be fully rendered. Each game will consist of four rounds of play, each round lasting one minute. The starting order of teams will rotate at the start of each round to guess the puzzle before the 30-second mark. If the team does not guess the puzzle within that time, each of the remaining team(s) will be able to guess once to solve the puzzle within a 15-second time limit.

## Requirements

1. Draw It or Lose It must be developed as a web-based game.
2. Each game will have the ability to have one or more teams involved.
3. Each team will have multiple players assigned to it.
4. Each instance of the game in play must have a unique game name. Similarly, each instance of the team in play must have a unique team name, and each instance of the player in play must have a unique player name.

## [Design Constraints](#_2et92p0)

1. The game must be developed in a programming language suitable for web-based applications.
   1. The game must be developed as a web-base game in order for The Gaming Room to scale their business and increase their customer base.
2. The game class must be able to add multiple teams, with each team connected via a distributed environment.
   1. Since the game is web-based, each team must be able to connect via a distributed environment.
3. The team class must be able to add multiple players, with each player connected via a distributed environment.
   1. Since the game is web-based, each player must be able to connect via a distributed environment.
4. Only one instance of the game can exist in memory at any given time.
   1. When the user selects the name for the game, team, and the player, the game application must check every instance of games, teams, and players that are in play to determine if the game/team name selected is not already in use.

## [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)

**"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.**

The UML class diagram above describes the system for the Draw It or Lose It game. The diagram shows seven (7) classes: Entity, GameService, Game, Team, Player, ProgramDriver, and SingletonTester.

The Entity class is a base class whose purpose is to hold common attributes (id, name) and methods (*getId()*, *getName()*, *toString()*) that are inherited by multiple child classes. The Game class, the Team class, and the Player class all inherit from the Entity class. The Entity class has a default constructor along with an overloaded constructor *Entity(long, String)*. The two attributes are protected to allow access by the *toString()* method in the three child classes (Game, Team, Player). Similarly, the default constructor *Entity()* is protected to allow access by the constructors in the three child classes.

Only one instance of the GameService class can be in memory at any time; the default constructor *GameService()* is private to implement the singleton design pattern. The GameService class can create instances of the Game class via *addGame()* method; the GameService object can have zero to many Game objects associated with it. Each created Game object is added to the *games* list. Before each Game object is created, an iterator pattern is used to check the name of each Game object in the *games* list to determine if the game name chosen by the user is in use or not, to ensure each game has a unique name.

The Game class can create instances of the Team class via *addTeam()* method; each Game object can have zero to many Team objects associated with it. Each created Team object is added to the *teams* list. Before each Team object is created, an iterator pattern is used to check the name of each Team object in the *teams* list to determine if the team name chosen by the user is in use or not, to ensure each team has a unique name.

The Team class can create instances of the Player class via *addPlayer()* method; each Team object can have zero to many Player objects associated with it. Each created Player object is added to the *players* list. Before each Player object is created, an iterator pattern is used to check the name of each Player object in the *players* list to determine if the player name chosen by the user is in use or not, to ensure each player has a unique name.

The Player class contains the overloaded constructor *Player(long, String)*. The Player class also contains *toString()* method which returns the Player information based on the Player object’s unique name.

The ProgramDriver class contains the *main()* method, which contains the code to drive and test the game application. The ProgramDriver class uses the SingletonTester class to validate the application’s singleton design pattern.

## [Evaluation](#_2o15spng8stw)

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | **Advantages**   1. macOS is a stable, high-performance computing platform for deploying server-based applications and services 2. Most secure option for web hosting due to: limited hardware, proprietary software, complex code 3. Most reliable option for web hosting due to robust hardware and software with minimal errors 4. Able to run free and open-source cross-platform web server software   **Disadvantages**   1. Closed source – NOT free to use 2. High cost 3. Updates are entirely controlled by Apple 4. Requires high-end price Mac hardware to run | **Advantages**   1. Open source – free to use 2. Low cost of development 3. Low maintenance 4. Does not require high-end hardware to run 5. Able to run Apache HTTP Server which is a free and open-source cross-platform web server software, and it allows software such as WordPress, vBulletin, and cPanel which can all be integrated with the server 6. User-friendly distribution such as Ubuntu   **Disadvantages**   1. Less support - not updated frequently 2. Less secure than Mac 3. Less robust hardware than Mac | **Advantages**   1. Only option that can run Microsoft-proprietary applications such as ASP, .NET, Microsoft Access, or MSSQL databases   **Disadvantages**   1. Closed source – NOT free to use 2. Updates dependent on Microsoft, typically once a week only 3. Requires high-end hardware (CPUs, high RAM) to run without server crashing – requires high-end price Windows hardware to run 4. Least secure option and requires frequent updates and each update costs money 5. Server becomes unavailable during the update | **Advantages**   1. Lowest cost compared to other options   **Disadvantages**   1. Least powerful compared to other options – least suitable option to host full-fledged server compared to other options |
| **Client Side** | **Advantages**   1. Supports most proprietary and open source applications, some that are not available on Linux 2. Requires less time than Windows due to cleaner UI with less clutters 3. Requires little expertise 4. Requires less time than Windows due to less setup and configuration needs   **Disadvantages**   1. High cost compared to Linux and Windows | **Advantages**   1. Open source – free to use   **Disadvantages**   1. Requires more time due to high configuration needed 2. High OS cost | **Advantages**   1. Very low cost compared to Mac 2. Cross—platform compatibility   **Disadvantages**   1. Requires more time due to additional setup and tool configuration needed compared to Linux and Mac | **Advantages**   1. Low cost 2. Low expertise required compared to other options   **Disadvantages**   1. Requires a lot of time since there are many variations of mobile operating systems depending on the type of mobile devices, especially Android platforms |
| **Development Tools** | * Requires knowledge of HTML, CSS, JavaScript, and Swift languages and frameworks * Requires IDE such as Visual Studio and Eclipse, along with tools such as GitHub and Netlify | * Requires knowledge of HTML, CSS, JavaScript languages and frameworks * Requires IDE such as Atom and Eclipse, along with tools such as GitHub and Netlify | * Requires knowledge of HTML, CSS, JavaScript languages and frameworks * Requires IDE such as Visual Studio and Eclipse, along with tools such as GitHub and Netlify | * Requires knowledge of HTML, CSS, JavaScript, and Swift languages and frameworks for iOS and Android * Development tools for iOS are similar to those for Mac |

## Recommendations

1. **Operating Platform**:

The Windows operating system is recommended to expand the Draw It or Lose It application to various computing environments. Windows operating system offers large user base and supports for many development applications and tools. It offers cross-platform capabilities with frameworks and libraries such as Angular and React.

1. **Operating Systems Architectures**:

The architecture of the Windows OS is a layered design with two main components: user mode and kernel mode. It is also a type of master-slave pattern, where the kernel mode is the master, and the user mode is the slave.

The user mode, also known as restricted mode, consists of two main subsystem groups: environment subsystems and integral subsystems. The three main environment subsystems are Win32, OS/2, and POSIX. Each of these subsystems run applications applicable to each type. The integral subsystems consist of security subsystem, workstation service, and server service.

The kernel mode, also known as privileged mode, manages the hardware and system resources, and allows code to be run in a protected memory area. It is made up of the Windows Executive services, kernel-mode drivers, and hardware abstraction layer.

1. **Storage Management**:

With the Draw It or Lose It game application being web-based, and based on the anticipated number of users and the size of the high-definition image library needed for the game, cloud-based storage solution such as the Microsoft Azure Cloud Storage is recommended. Azure offers flexible purchasing and pricing options to include an option to pay only for what was used. In addition, Azure offers support for various open-source frameworks and platforms.

1. **Memory Management**:

Windows OS uses both physical and virtual memory. For each process which will run on the physical memory, the operating system first commits it to the virtual memory. If the operating system’s physical memory is sufficiently large to accommodate all processes requiring memory, the processes are mapped to the physical memory. Otherwise, the system’s virtual memory manager loads each process on the physical memory sequentially – as one process ends, a new process gets written to the physical memory, until all processes that are committed to the virtual memory are executed.

1. **Distributed Systems and Networks**:

Communication between platforms for the Draw It or Lose It game application can be accomplished by implementing client-server system (and RESTful API) – the system uses distributed software and network resources to provide interaction between clients and the server. This allows many clients to share the resources provided by a single server. Each client requests input from the user (in this case, the game players), then requests pertinent data from the single server. The server then analyzes the request from each client and returns the requested data back to each client. The non-critical functions and user interface concerns are moved to the client-side, which allows the server to focus only on the critical processes. Since each client request to server is independent of each other, each client’s connection to the server is also independent from other clients’ connections. However, when there is a server outage, all connected clients are affected.

1. **Security**:

The Windows operating system offers Microsoft Defender Antivirus with each system which offers real-time, always-on antivirus protection. Microsoft Firewall is also available which helps to prevent unwanted intrusion through the internet or a network. There are also plethora of third-party antivirus solutions available, such as Norton and McAfee.

Security is an integral part of the RESTful API styles. For the Draw It or Lose It game application, the server can utilize a Basic Authentication method (or a more robust security method) to secure the web API. To do this, an authenticator is used to verify valid username and password credentials of the players. In addition, the Authorization method can be implemented to restrict certain program methods to only the authorized and authenticated users, such as the administrators. This will help to ensure that each user is provided with the least amount of privilege necessary, with the goal of protecting and securing the protected data from unwanted intruders.