

The Gaming Room

# **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 | 05/26/2024 | Chandler Millican | Creation of software design template |
| 1.0 | 06/11/2024 | Chandler Millican | Filled out Domain Model Section |
| 1.0 | 06/21/2024 | Chandler Millican | Filled out Recommendation Section |

## [Executive Summary](#_sbfa50wo7nsh)

We have been hired by The Gaming Room to develop a web-based version of the gaming app, Draw It or Lose It. The biggest restriction in this development is going to be designing it to only allow one instance of a game, team, or player at any given time.

## 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)

* The first constraint is going to be the size of the game. Building a game in a web-based environment means that the game will not be able to be as large as a game that uses its own engine.
* Hardware will also be a constraint. Since the client wants the game to be run on a large range of hardware, from mobile to desktop, the team will have to work harder to optimize the software for each different kind of hardware.
* Bandwidth and budget constraints are a concern because the program will have to support multiple unique teams and players all in a single game instance.

## [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 Entity class in the UML Diagram serves as the parent class for the Game, Team, and Player classes. This means they will inherit the data members of the Entity class.
* The GameService class keeps a list of the games created and accesses the Game class to be able to get the name and id of the game and add it to the list. This class also allows the user to get a count of the games created. It also includes functions to iterate through the Team and Player list to look for duplicate ids.
* **The Game and Team class work the same way because both create a list that can be checked later by the singleton tester to see if there are duplicate entries. The Game class creates the team list while the Team class creates the player list.**
* The Program driver is the class that will contain our main function, therefore will be the class that drives our program. It will use the SingletonTester class to check for duplicates in any of the lists created with 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** | **Characteristics:**  Known for being stable and reliable  **Advantages:**  Simpler and more streamlined user interface than other OSes  **Weaknesses:**  Software tools and application are not as readily available as other OS like Windows | **Characteristics:**  Open source and able to be flexible and highly customizable  **Advantages:**  Free to use which can save on cost for consumers when they switch to Linux  **Weaknesses:**  Not user friendly and requires a deep understanding of the system | **Characteristics:**  Most common OS and very customizable and user friendly  **Advantages:**  Maintains backwards compatibility which allows older software to run on the new version. This is necessary when dealing with legacy applications.  **Weaknesses:**  Dependent on updates for security patches and bug fixes | **Characteristics:**  Can be used as a personal webserver, but are not equipped to handle multi-user serving  **Advantages:**  Mobile OSes can support multiple apps open at one time, making them ideal for multitasking  **Weaknesses:**  Mobile devices are more susceptible to security threats, so they will need regular updates against viruses and malware. |
| **Client Side** | **Considerations:**  The macOS user interface is user friendly and minimizes the user learning curve. Because of this macOS is enticing to clients and with multiple clients you must consider elevated development cost, time, and the different levels of expertise needed. | **Considerations:**  Client-side development on Linux has similar web technologies as Mac. Compatibility test will need to be conducted across Linux distributions | **Considerations:**  Licensing cost for Windows may be higher than other operating platforms and open-source alternatives. This will impact on the overall project budget. | **Considerations:**  A key consideration for mobile development is designs for different screens and how to handle connectivity issues. |
| **Development Tools** | **Languages and Tools:**  Commonly used languages for MacOS development are Swift, Python, Java, and C#.  Common IDEs used are Visual Studio Code and Xcode. | **Languages and Tools:**  Commonly used languages for Linux development are Python, Java, C++, and C.  Common IDEs used are Visual Studio Code, Eclipse, and PyCharm. | **Languages and Tools:**  Commonly used languages for Windows development are Python, C++, Java, and JavaScript.  Common IDEs used are Visual Studio, PyCharm, and Eclipse. | **Languages and Tools:**  Commonly used languages for Mobile Device development are Swift, Java, and C#.  Common IDEs used are Android Studio and Xcode. |

## 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**: The operating platform that I would recommend The Gaming Room start with is Windows. It already has a large customer base and is widely used across the globe. This makes it a good choice for software development because of the familiarity and expertise many individuals already have with Windows. Since most developers are already familiar with the platform it will reduce the learning curve to use the platform and enable faster development cycles. The large customer base that is already in place also ensures that when the game is rolled out it will reach, and its accessibility will be greatly increased compared to operating systems such as Linux.
2. **Operating Systems Architectures**: A key advantage of web-based software is its compatibility with a wide range of Windows platform systems and mobile platforms. That is because their platforms already have the necessary capabilities and resources to handle the workload of the software. Since it is web-based the user will be able to access the software from several devices such as a laptop, desktop, or mobile device without requiring anything specific or specialized configurations. This means the user will be able to access the software through their preferred browser without installing additional software or plugins.
3. **Storage Management**: Since our client is looking to develop a web-based application, I would recommend cloud-based storage. This will allow for information to be easily accessed from anywhere if a connection with the cloud can be made, making it extremely beneficial for mobile platforms.With Windows OS you also have several features available to you for storage, such as Storage sense. Storage sense allows you to evaluate and manage files on your hard drive and the amount of space used.
4. **Memory Management**: A database or library will need to be created to hold all the pictures for the game. This will allow the pictures to be stored in a secure area with the rest of the project outside the default picture folder.
5. **Distributed Systems and Networks**: Since every operating system is different we need to find ways to publish the game so that it can run on all devices. This is where engines such as Unity and Unreal come in, they are examples of IDEs that enable cross-platform game creation and can be run on any device. Once the game is created the file can be exported to the web, iOS, Android, and many more options that allow cross-play. The client would need to ensure that they have strong backups for their server and that the server is strong enough to support a large player base, so they can avoid connectivity issues and other issues such as outages.
6. **Security**: Windows comes equipped with built-in security protection for software. The system will automatically scan for malware, viruses, and security threats. It will also update automatically, since security threats are always changing, so that the system and user information remains protected. In addition to that they offer user account control settings that the user can customize to protect data even further.