

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.3

## 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 | 11/11/2023 | Douglas Bowers | Introduction of new classes, additional requirements |
| 1.2 | 11/26/2023 | Douglas Bowers | Added to evaluation and recommendations |
| 1.3 | 12/8/2023 | Douglas Bowers | Additional information added to development and 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 would like to develop an app called, “Draw It or Lose It” which is inspired by the 1980s television game “*Win, Lose or Draw*.” They want it to be accessible on iPhone and Android devices and want to have team capabilities with additional abilities to have multiple teammates.

## 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)

* Game must accommodate multiple teams and assign multiple players to each team
* Needs to support iPhone and Android devices
* Input validation for new and returning players
* Input validation for new and returning teams
* Only one game can exist at one time
* Check for duplicate names, teams, and game instances
* Must have adequate security measures for user data and servers
* Must have enough storage for photos
* Servers must be able to handle player base

## [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 base class is created to hold together the contents of the three subclasses: Game, Team, and Player. It contains the common attributes, names, behaviors, and strings for them to function. The subclass Player extends Entity, which holds information about each player in the game and outputs that information. The subclass Player also has a relationship to the Team class, which is where each team will have multiple players assigned to it. Each team holds a private list of players and a public method to add additional players. Like the Player and Team relationship, the Game subclass also holds a relationship with Team. A private list of teams and the ability to add additional teams is held within the Game class. An iterator pattern is implemented to verify that each team name is unique and duplicate names cannot be added. The GameService class also has a relationship to the Game class, as it holds a private list of games. This class can add games and check that each game name is unique using an iterator pattern. To verify that these conditions are set, the SingletonTest class is implemented to utilize the ProgramDriver class. This tests that all software requirements are met for the game application by utilizing standard object-oriented principal standards such as inheritance, overrides, relations, and singleton and iterator patterns.

**"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** | Mac offers server- based environments. Going with a MacOS server can offer many major advantages for mac clients on a network. This would include full support for the Mac application, as well as general administration access that includes a great graphical interface. The downside, however, would be the cost. It would be expensive to maintain. It is also not great for companies that need to rely on third party software and programs. | Linux offers a lot when it comes to web hosting. Some of the largest companies, such as Google, use Linux servers for their services. One of the biggest advantages of Linux would be the free, open-source servers. This means that deployment and use of tools would be either very cheap or even free. Since Linux is extremely customizable, many companies offer vast security features that allow for customization of security preferences. Linux web hosting also supports Python, PHP, Perl, and Ruby. However, the downside of Linux would be that experience is needed. If you are unfamiliar with the system, it can be challenging to understand, leading to additional training and costs. Some applications may also not be supported. | Windows offers a well-rounded experience. It is proprietary software which requires a license to be paid for. While this can be expensive, Windows offers many great advantages for using their webhosting. Many applications and third-party software are available, updating software and hardware is streamlined, and there are many familiar coding languages supported and offered through Windows. | While mobile device servers are not well known, or practical, it is possible. Companies such as Oracle offer mobile sever-side implementation that lets users manage applications, other users, devices, and data through mobile or remote databases. Oracle also supports both iOS and Android dev tool supports, and synchronization to NoSQL. |
| **Client Side** | Mac offers a wide range of well supported web browsers that feature various tools for developers. Cross-browser testing is also made simple, and it offers average development time and deployment. However, you are limited to Apple products with MacOS, or partitioning your device to run MacOS. | Linux offers a wide range of well supported web browsers with various tools for developers. It also works on every web browser due to the large amount of open-source software available. It uses a tree file structure and can lead to faster development times and deployment. | Windows offers a wider range of well supported web browsers that feature various tools for developers. It also offers easy cross platform testing but can be more difficult when working with MacOS. Since it is widely used and supported, development times and deployment will be fast. | Android app development offers expertise when it comes to tools and available software. However, it can be difficult when testing other environments and browsers, leading to longer development times. |
| **Development Tools** | MacOS offers documentation deployment. It can also run both Windows and Linux through local Virtual Machines. Apple requires applications to be reviewed and approved which can lead to wait times. Apple also charges $99 per year for their developer program. Eclipse for Java is also available to use.  Swift can also be utilized with multiple IDEs such as Atom. | Linux offers deployment at any time. It can also easily run Windows and MacOS through local Virtual Machines. Linux does not require a license cost to use. Development is streamlined due to open-source features, as well as Shell prompt and terminal programming. Eclipse for Java is also available, as well as Atom. | Windows offers deployment at any time. It also requires extensive documentation for deployment on Windows. It can run Linux through Virtual Machines. No licensing costs for development. Visual Studios is available for Java and HTML, as well as Eclipse for Java. | Xcode 12 for deployment to iOS, as well as a $99 a year Apple developer program for iOS. Swift UI is also available. |

## 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 personally recommend Windows:

* Integration with current builds of Android, cross-platform solutions, easy development environment setup
* Allows for developers to design, develop, and deploy apps and solutions using the Microsoft suite
* Various options when it comes to emulation
* Largest userbase, popular operating system for game development, and vast amount of third-party support

1. **Operating Systems Architectures**:

Windows offers the separation of its user mode and kernel mode. This means that the user mode processes user interactions and affects what the user experiences. The kernel mode deals with inputs and outputs, memory management, networking, hardware management, and routines. Windows also utilizes a directory structure to hold data. Multiprocessing and modularity are also supported to allow for customization of the system.

1. **Storage Management**:

Windows 11 allows the user to freely manage files on the hard drive, giving you the ability to manage your storage with ease. It also allows for device management from any connected device. Additionally, Microsoft also offers many forms of cloud-based storage which is secure, supported, and reasonably priced.

1. **Memory Management**:

Windows 11 can utilize the entirety of its virtual memory address space, which would be more than enough for this application. Windows is constantly being updated and improved for its memory management, allowing for faster and more efficient loading.

1. **Distributed Systems and Networks**:

Going with Microsoft cloud-based services would provide ease of access when dealing with distributed system and network. It would also allow for cross-platform functionality, in addition to collaborations with other games/systems. This would ensure that the game can run on multiple platforms using a single game instance.

1. **Security**:

Microsoft offers built-in security features and firewalls, such as Windows Defender, to help protect the system as well as any data stored within. Cloud servers also offer additional security using VPN storage, which would be available within the cloud, further enhancing security, as well as the addition of the protection of user data through whitelisted server access.