

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template**](#_l6ti7uoag22u)1

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

[**Document Revision History**](#_grjogdjh5fi8)2

[**Executive Summary**](#_sbfa50wo7nsh)3

[**Design Constraints**](#_2et92p0)3

[**System Architecture View**](#_ilbxbyevv6b6)3

[**Domain Model**](#_8h2ehzxfam4o)3

[**Evaluation**](#_2o15spng8stw)3

[**Recommendations**](#_m8aleynsvzvc)5

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | <03/20/22> | Jonathan Wolanyk | In this revision, I completed the “Executive Summary,” “Design Constraints,” and “Domain Model” sections of this document. |
| 1.1 | <04/03/22> | Jonathan Wolanyk | In this revision, I completed the “Evaluation” section of this document. |
| 1.2 | <4/16/22> | Jonathan Wolanyk | In this revision, I completed the “Recommendations” section of this document. |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room company is seeking to create a web-based game that will serve multiple platforms based on their current game, Draw It or Lose It, which is only available as an Android application presently. The game must be able to have one or more teams of players that can guess a puzzle’s solution based on images that render to completion at a steady rate over 30 seconds. The solution will be a phrase, title, or thing based on the provided image. If the current team does not complete the puzzle within the allotted time, the remaining teams can offer one guess each to solve the puzzle within a 15 second timeframe. Team names and game names must be unique, and the system must allow users to check whether a name is in use when choosing a name for themselves or their team.

## [Design Constraints](#_2et92p0)

* The application must be compatible with all web browsers. This means that our development team must test and ensure the game’s compatibility on all browsers, such as Mozilla Firefox, Safari, Google Chrome, Opera, etc.
* The application must be able to scale with a given browser’s window size. This means that we must ensure that information is not cut off by a given browser’s dimensions.
* The game must be able to have one or more teams involved. In other words, the game must run in situations involving only one team and in situations involving multiple teams.
* Each team must be able to have multiple players assigned to it. As a result, teams must be able to properly form with a single player or multiple players.
* The game can only have one instance exist in memory at a given time. This means that the team must create unique identifiers for each instance of a game, team, and player.

## [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 ProgramDriver class will house the main() method for the game. The main() method will use the SingletonTester class’s testSingleton() method to check whether only one instance of the GameService exists at the time of launch.

The GameService class is responsible for housing the list of Game objects. The method getInstance() in this class is a Singleton Pattern that ensures only one instance of a GameService object exists at the time of launch. If a GameService object does not exist at launch, the getInstance() method creates one. The GameService class also houses a method that can add Game objects to the current list of games.

The Game class relates to the GameService class through multiplicity. For each GameService object, there can be zero to many Game objects. The Game class is responsible for creating new games and adding to the list of Team objects housed within this class. The Game class relates to the abstract Entity class through inheritance. The Game class inherits the id and name attributes from the Entity class.

The Team class relates to the Game class through multiplicity. For each Game object, there can be zero to infinite Team objects. The Team class creates new Team objects as well. The Team class is also responsible for housing and adding to a list of Player objects. The Team class inherits the id and name attributes from the Entity class.

The Player class relates to the Team class through multiplicity as well; for each Team object, there can be zero to infinite Player objects. The Player class is also responsible for creating new Player objects. Like the Game and Team classes, the Player class inherits the id and name attributes from the Entity class.

**"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** | The Mac operating system is known for being less susceptible to security issues and vulnerabilities. Mac also has a built-in terminal, which provides easy access to tools such as git and virtualization software. Additionally, Mac applications are integrated with other Apple products. A drawback of Mac development is that its creation and hosting are quite expensive. | Linux is open source, so there are more hosting options than other operating systems. The main benefits of Linux web hosting are its compatibility with most popular web hosting software and hardware, and cost effectiveness. The most significant drawbacks of Linux are the incompatibility with Windows applications and the higher level of expertise to develop than Mac and Windows. | Windows is the most widely used operating system in the world, which results in a larger pool of developers being able to work on the system after it has been released. The main benefit of Windows is the continued support of one’s web design program. Windows is weak when dealing with high-volume websites and has a higher cost to maintain than other options. | Mobile devices are useful for hosting a web-based software application because there is no need for installation and development costs are lower than on more traditional operating systems. The main disadvantage of development for mobile devices is ensuring that the application scales properly to all device dimensions. |
| **Client Side** | Mac clients are easier to support because of the integration across Apple products. Whether clients are using an iPad or MacBook, they will be supported with little time or additional effort required by the developers. This leads to less time and expertise needed to develop web-based software using the Mac operating system. | The Linux operating system can host a more stable environment, give more control to developers, and is more affordable to host, than its counterparts. Linux applications face compatibility issues with many client devices (namely Windows and Mac devices) and require third-party software to run properly in many cases. | While Windows requires less expertise to work within, is quite expensive to develop and maintain. There are far more Windows users than Linux or Mac, so fewer issues should arise from users if a web-based application utilizes the Windows operating system. | Mobile devices are less flexible when processing applications that are developed for other platforms. Developing and maintaining web-based applications for mobile devices is less costly than the alternatives while being more difficult to maintain in terms of ensuring each type of mobile device is supported by the application. |
| **Development Tools** | The main programming languages supported by Mac are Swift, C, C++, Python, Java, Ruby, AppleScript, JavaScript, and C#. Some popular IDEs supported by Mac include PyCharm, Xcode, Visual Studio, Atom, IntelliJ, and Eclipse. | Except for Windows-based languages, Linux supports almost every programing language. The most popular IDEs for Linux include Atom, Visual Studio Cost, Eclipse, Apache NetBeans, IntelliJ, and Spacemacs. | C++ is the most popular choice for Windows development. C#, JavaScript, TypeScript, Python, and other .net languages are also used. Popular IDEs for Windows development include Visual Studio, PyCharm, IntelliJ, and Eclipse. | Java is the most popular language for mobile development. Kotlin is the next evolution of Java and is quite popular as well. Android Studio, Xcode, Eclipse IDE, Visual Studio, and Atom are popular IDEs for mobile development. |

## Recommendations

1. **Operating Platform**: The most appropriate operating platform for The Gaming Room’s Draw It or Lose It program would be Linux. The primary advantage Linux has in expanding a program to other computer environments is its compatibility with the most popular web hosting software and hardware. Because Draw It or Lose It will be developed as a web-based application, Window’s incompatibility with Linux programs is negated. In addition, from the client-side, Linux can host a more stable environment, which gives more control to developers with a lower hosting cost than its competitors. From a business perspective, Linux is also much more cos-efficient than macOS or Windows. Linux is free and open source, so much more information is freely available. Additionally, there is no difference between the cost of desktop or server licenses (unlike Windows). This is a great benefit when considering which operating platform to utilize for the launch of a game such as Draw It or Lose It.
2. **Operating Systems Architectures**: The basis of Linux development revolves around the kernel and shell. The Kernel is the core program of a computer’s operating system; it has complete control over everything in the system (GeeksforGeeks, 2019). For example, the Kernel is responsible for file management, process management, I/O management, and memory management, among other tasks. The shell is a program that provides an interface for users to utilize operating system services; it can take inputs from various sources. The shell is often broken into two categories; the “Command-Line Shell” and “Graphical Shells.” For example, in Linux, the “Terminal” program is the Command-Line Shell. Comparatively, the Command Prompt is the Command Line Shell of Windows. Users mainly interact with Graphical Shells, which allow users to open, close, move, and resize windows. Linux’s Korn Shell (KSH) is the newest and most powerful Command-Line Shell, so it would be wise to utilize it to develop Draw It or Lose It.
3. **Storage Management**: Linux-based servers are quite helpful for businesses of all sizes for storage management; It is the only operating system used on TOP500 supercomputers (OpenSource.com, 2018). Linux provides a secure foundation for applications and flexible integration with legacy management; this can be especially useful for future Draw It or Lose It iterations. Linux-based servers also provide management tools for individual systems via an intuitive web interface. As a result, The Gaming Room’s staff will be able to easily take on the task of storage management once our team completes the Draw It or Lose It project.
4. **Memory Management**: Linux’s memory management capability is well-respected because it supports virtual memory. Virtual memory alleviates the stress placed upon random access memory (RAM) while a device is running programs. By utilizing Linux to develop Draw It or Lose It, the program will run smoothly on users' devices compared to operating systems that do not have as efficient memory-management capabilities as Linux. Another benefit of Linux in terms of memory management is its demand paging; this process loads the contents of an executed command on demand rather than storing unnecessary information into RAM.
5. **Distributed Systems and Networks**: The client-server model is essential to implementing Draw It or Lose It; Users will be requesting information from the server regarding various information such as requesting leaderboard information, setting their high score, logging in, etc. The client-server model is much more secure and stable than utilizing a peer-to-peer connection since The Gaming Room will want to maintain control over who can access and manipulate information at any given time. In addition, distributed systems are the optimal solution for scalability and reliability since an entire network of nodes supports a given system. If a single node were to fail in a distributed system, the rest of the nodes would be able to cover while the malfunctioning node is repaired. Since Draw It or Lose It should be able to be played at any given time, the consistency provided by distributed systems and networks is essential to the game's success.
6. **Security**: All operating systems can be compromised. However, some operating systems are more secure than others. Linux, for example, the recommended operating platform for Draw It or Lose It, has segmented working environments to secure its platform from attacks. Linux also has a smaller userbase than Windows or macOS, making the population of intruders much smaller and requiring a higher barrier to entry since less information is published about the platform that would aid potential unauthorized users from penetrating The Gaming Room’s servers. In terms of general security, the client-server distributed application model ensures that users are only given access to information that they are permitted to access. The server controls the flow of information; This protects users’ data by only distributing data to users who are allowed to receive the information. It also protects The Gaming Room by controlling which users can change data on the program’s database.

References

GeeksforGeeks. (2019, January 24). Linux Tutorials | Linux Architecture | GeeksforGeeks [Video]. YouTube. <https://youtu.be/maupflAJdQ4>

OpenSource.com, et al. “What Is a Linux Server and Why Does Your Business Need One?” Linux.com, 31 May 2018, https://www.linux.com/news/what-linux-server-and-why-does-your-business-need-one-0/.