

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 | 09/14/2022 | Oggie Hall | Added executive summary, design constraints, and domain model explanation. |
| 1.0 | 09/28/2022 | Oggie Hall | Added evaluation |
| 1.0 | 10/11/2022 | Oggie Hall | Added recommendations |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room company would like to develop a web-based gaming application that serves multiple platforms. The game, Draw It or Lose It is currently available as an Android app. The game consists of teams competing to guess what is being drawn. The application will render images from an extensive stock library, and the players may guess the drawing using the clues. There will be four rounds lasting one minute each. The clues are given at a constant rate, and the picture is complete in 30 seconds. The remaining teams are allowed one guess each to solve the puzzle in 15 seconds if the active team does not guess correctly in time.

## [Design Constraints](#_2et92p0)

* Multiple teams
* Multiple players
* Game and team names must be unique and allow users to check if already in use
* One game instance is allowed to exist in memory at any given time.
* Stock images potential copyright issues

The development team must know web-based application development principles to migrate the existing Android application. Presentation is critical; therefore, an understanding of web design is also recommended. This would allow developers to provide the best user experience to the players. The most crucial step would be to verify all required packages and plugins powering the mobile app are available for the web-based version. Any new images added to the game will need to be licensed or copyright free.

## [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 UML diagram below displays a visual design of the game program. The Entity class joins the Game, Team, and Player classes meaning each would inherit its attributes. This is indicated by the arrow pointing all three classes to the superclass. The ProgramDriver class has an arrow pointing to the SingletonTester class, which means it would utilize its method to test code for a single instance. This would meet the requirement of having one game instance in memory. The GameService class holds many attributes and methods as it will provide a lot of the core functionality, as shown by the line connecting to the Game class with a multiplicity of zero to many. The Game class is connected to the Team class with a multiplicity of zero to many. The Team class is connected to the player class with a multiplicity of zero to many. By following this diagram, the coding process is made more efficient as you essentially now have a map to navigate to your intended destination.

**"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** | Apple discontinued macOS Server in April 2022. A few of the popular features were made available with every installation of macOS. Caching server, file sharing server, and time machine server are a few. | Low-cost and open source. You will need someone familiar with Linux server. Not as exposed to viruses and attacks like windows due to its low market share. Software compatibility may be an issue. | Best support for servers. Setup could be costly but will be fully functioning. Most popular OS for servers so easier to find users to manage. Software compatibility high and expertise available. | Server is not a mobile device and requires power and connectivity above the average mobile device. No major mobile server software or hardware exists. |
| **Client Side** | Moderate expertise and time required. Similar cost to Windows. Able to program languages and test in compatible web browser supported on this platform and mobile devices. | Maximum expertise and time required. Low cost as software is typically free or low cost. Linux is not commonly used so it could face compatibility issues. | Moderate cost and time required. Many resources exist that could aid or guide setup. Time dependent on the level of experience with windows operating system. | Moderate cost and no time. Applications can be downloaded and installed. Users can open web browser to visit game. Multiple operating systems, multiple mobile devices would need to be tested. Device performances vary. |
| **Development Tools** | Swift is the most popular language on the platform to write applications. XCode, Visual Studio Code, and Atom are some of the best code editor applications on macOS. | Atom is a text editor for programmers that is free to use and open source on all major Linux distributions. Eclipse is primarily used for Java but supports other languages. | Visual Studio code can be used for free. It is a code editor redefined and optimized for building and debugging modern web and cloud applications. Languages supported from JavaScript, HTML, C, to java, python, and C#. | For iOS, the development tools are like MacBook. Swift programming language applications so that iOS and macOS are compatible. Android Studio is an app builder with an integrated development environment (IDE) optimized for Android apps. |

## 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**: Windows is the operating system that I would recommend to the developers at The Gaming Room. Android Studio is available on this platform and can be used to analyze the current build. Since windows is the most widely used operating platform, each developer should have a working knowledge of the platform. There are many IDEs, emulators, and databases that exists because of the large userbase.
2. **Operating Systems Architectures**: Windows architecture consists of two main components, user mode, and kernel mode. The kernel mode is also known as the privileged or master mode. This mode processes the system's management of memory, networking, hardware, inputs and outputs, and scheduled tasks (routines). User mode processes graphical user interfaces and affects systems that the user interacts with via input device. Windows uses a directory structure to store data. Windows support multiprocessing and modular hardware access to allow for system customization.
3. **Storage Management**: For Draw It or Lose It, we recommend utilizing a cloud storage service like Microsoft Azure. Competitive prices, 24/7 customer support, and continuous updates are a few of the benefits of this storage type. Cloud based storage provides a way to scale up or down depending on the size of the user base. Disk management and storage sense are built-in tools to help maintain and manage storage devices connected to the system. Cloud storage does not take up physical space, so files are backed up and secured online.
4. **Memory Management**: Windows has built-in memory management. This is accomplished through disk paging to act as an extension of the physical memory or RAM. Disk paging reserves a partition of the hard drive disk to store temporary files. Processes are separated into smaller tasks only to be loaded into memory when it is required for immediate processing. The game database or library of images should be readily available so that users can play simultaneously.
5. **Distributed Systems and Networks**: We recommend using a client-server distributing system to provide gaming services to the players. Each client application would depend on its connection to the single server application for Draw It or Lose It. For mobile devices we can provide an application to the app store. For macOS, Windows, and Linux devices we can provide access through an updated web browser. The company must provide immediate access to the game server as well as a backup server to support a large player base. If connection is lost, information can be restored using temporary storage or recovered from backup.
6. **Security**: Due to the client-server structure security and data protection is provided by The Gaming Room company. Any data the user shares can be uploaded and stored in the cloud. The cloud provides encryption methods to protect user data. Players can provide username and passwords. Additional verification methods can be implemented depending on the account set up device. If the user is on mobile, we could allow them to use biometrics security. If the user is on mac or PC we can provide multifactor authentication, when a code is sent to their designated device and must be entered along with the password. Since the server hosting the game is accessed remotely, users do not need to scan for malware, viruses, or any other attacks.