

“Draw It or Lose It” Game Application

# **CS 230 Project Software Design Template**

Version 1.2

## 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  1.1  1.2 | 09.18.2022  10.02.2022  10.14.2022 | Scott Dixon  Scott Dixon  Scott Dixon | Update code in application and define goals and constraints of operating system functionality.  Add evaluation of various platforms.   1. Added Recommendations for **Operating Platform:** Recommend an appropriate operating (server) platform that will allow The Gaming Room to expand Draw It or Lose It to other computing environments. 2. **Operating Systems Architectures:** Describe the details of the chosen operating platform architectures. 3. **Storage Management:** Identify an appropriate storage management system to be used with the recommended operating platform. 4. **Memory Management:** Explain how the recommended operating platform uses memory management techniques for the Draw It or Lose It software. 5. **Distributed Systems and Networks:** Knowing that the client would like Draw It or Lose It to communicate between various platforms, explain how this may be accomplished with distributed software and the network that connects the devices. Consider the dependencies between the components within the distributed systems and networks (connectivity, outages, and so on). 6. **Security:** Security is a must-have for the client. Explain how to protect user information on and between various platforms. Consider the user protection and security capabilities of the recommended operating platform. |

## 

## [Executive Summary](#_sbfa50wo7nsh)

The design of the latest application will add new functionality and fun to the already popular “Draw It or Lose It” game adaptation for the Apple IOS and Android operating systems. Creative Technology Solutions is proud to work toward completion of the latest version of the game using agile philosophy, adopting sound coding techniques and reducing the time to market by utilizing the modern design implementation of Maven and advanced application programming interfaces.

## [Design Constraints](#_2et92p0)

The game will allow one or more teams to be involved. Each team will have multiple players assigned to the game-space. The names of the teams and names of the game will be unique so that the application will be running a new game each time. Only one instance of the game will exist in the application’s memory at any given time. The added constraints are as follows:

* The application must run in cross-platform mode so that players can use both Apple IOS and Android devices.
* The application must be able to operate with many users and allow for numerous yet unique login passwords.
* The application must allow the use of security systems which are in place for either face recognition or fingerprint access to the main game application.
* The programming language will be Java since it can be used in cross-platform applications and has the security features necessary for running this type of game application.

## [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 design outlines the steps which need to be followed in the operational process of running the game application. The Program Driver Class holds the main method. Program Driver uses the Singleton Tester to test of there is already an instance of Game Service.

Entity class is the parent class to the Game, Team, and Player classes. Game, Team, and Player will all inherit attributes from the Entity class. Game Service by the virtue of the Singleton class will have only one instance of each game running at a time. Each Game will only have one unique Team at any time. Each team can only have one of a unique player at one time.

**"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)

The client has specified that the game, Draw It or Lose It, be programmed to run only on mobile devices using the Apple IOS or Android OS. Therefor the application will not be offered to users of Linux or Windows at this time. The design philosophy behind this decision works well with the sales projections of both the Apple store and the Google App distributor.

The added ability to use the application for “Draw It or Lose It” in an expanded server-side environment will allow the number of users to scale to thousands of players. The cost is within project guidelines and will require an added two months of build time. The project is workable from a cost perspective and will only require an additional short build time from the developers viewpoint.

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Updated configuration ability for Server-Side applications based on Apple IOS.  This application will be offered at a later time. The application will be in development until that time. | This platform allows for updated scaling of Server-Side application.  The application will be offered in Linux and Ubuntu since the operating system is adaptable and can host cross-platform systems. | This platform allows for updated scaling of Server-Side application.  Windows servers will be used, and the lower cost will promote the game application to be widely accepted. | Mobile devices will not be targeted for use since the memory limitations and cost of hosting a mobile server-side platform will hinder design applications. |
| **Client Side** | The Mac system is ideal for mobile application development. The dev fee is only $99 a year and with that, the company can develop many versions of the game application.  The cost is within project guidelines and will require an additional two months of build time. | Linux development is extremely adaptable and cost effective. The system can use Java, C# and all of the necessary programming languages needed to bring the application to market.  The cost is within project guidelines and will require an additional two months of build time. | Windows will be a good tool to design and test the “Draw It or Lose It” game app. The cost is within project guidelines and will require an additional two months of build time. The IDE is excellent and can support the necessary languages to bring the game application to completion. | The use of REST API and the fact that the application will run on Apple IOS and Android platform allows the developers to complete the program using two -week duration sprints and application updates every three months.  The focus of our client will use mobile devices as the main method of software design in the marketplace. The relative cost of the mobile designs will allow the user to play at any time and at any location. |
| **Development Tools** | The application will be built using Swift and IOS cross-platform Xamarin. | The application will be built in Linux and the cross-platform mobile applications. | The relevant IDE will be VS Code and Eclipse to design the Java programs necessary to run the game application. | The mobile game application will use Xcode and Java along with C# in the software design. |

## Recommendations

We recommend the use of the following design specifications to maximize the cost effectiveness and durability of design. The

1. Operating Platform: The mobile operating platforms will be the primary design base. This is the platform that will be marketed at this time. The cost of the platform and the possibility for expansion makes the feasibility of this design plan possible.

* Operating Platform UPDATE: I recommend the utilization of Cloud Server Technology including:
* Cloud servers such as Amazon Elastic Compute Cloud, Microsoft Azure and Google Compute Engine.
* Private cloud servers. A cloud server may also be a compute instance within an on-premises private cloud.

1. **Operating Systems Architectures**: The platform for the game will be the Apple IOS and Android operating system. These will be able to deliver the game to players around the world. The market share for mobile operating systems is shown below. The number of Apple IOS and Android OS make the project viable and later the plan will be to implement the game “Draw IT or Lose It” to the PC market.

* Operating Systems UPDATE: I suggest the operating platforms for the new release of the game “Draw It or Lose It” to migrate to a cross-platform release including:
* Apple IOS
* Android OS
* Windows OS
* Mac OS
* Linux OS

Graphical user interface

Description automatically generated with medium confidence

1. **Storage Management**: The storage management of the typical mainframe as well as the mobile OS is adequate for the implementation of the game software. The virtual memory used will be in the range of .5 GB to .75 GB for the application.

* The Cloud implementation will allow the game application to run flawlessly with Gaming Databases
* The Google Cloud’s database options will power the back end, run leaderboards, manage player authentication systems, and more.

1. **Memory Management**: Both proposed operating systems, Apple OS and Android OS, will be using the off-platform storage in the cloud API and use no more that .75 GB with virtual page swap and this will be more than sufficient for memory allocation.
2. **Distributed Systems and Networks**: Knowing that the client would like Draw It or Lose It to communicate between various platforms the game developers learned about Azure and cloud-based storage systems applies to gaming performance to manage concurrency with low latency and big spikes in demand.

**6. Security**: The need for security is outlined using the Java runtime and execution models that the game program is based on. The use of the Singleton design model is paramount in achieving the application which allows the player to take part but doesn’t allow more than one login of the unique user at a time. The application will be backed up with Azure web security and the application will be safe from hacking since it is built upon WEB 3 principles of login authorization.

* We are suggesting the implementation of Cloud Services and the Amazon Inspector which is a vulnerability management service that continually scans AWS workloads for software vulnerabilities and unintended network exposure.
* First, we recommend taking the time identifying which cloud products and services are being used in your organization, and which ones might be considered in the future. Then, you’ll need to assess and audit those items, analyzing their security and potential vulnerabilities.
* Within each application, the privacy and security settings will be available in the cloud security system panel. To understand which settings are available and take full advantage of them to grant “Draw It or Lose It” the highest possible level of security.
* In many cases, “Draw It or Lose It” cloud services will need to take extra efforts to prevent data loss and preserve data integrity by encrypting your data and securing your connections. It’s the responsibility of the cloud-based security software to allow legitimate network traffic and block suspicious traffic.
* Cloud applications allow you to reduce the amount of physical mainframe you maintain, but you will still be accessing data and services with specific devices. You’ll need some way to manage and monitor those devices.
* We suggest using a multi factor authentication model to insure against data loss and cloud computing integrity.
* Similarly, you’ll need to consider user-level controls. Establish varying levels of user permissions, to restrict access to your most valuable or sensitive information.
* For “Draw It or Lose It” It is also important to monitor cloud activity from a high level, and report on that activity so you can better understand your risks and ongoing operations.