

Draw It or Lose It

# **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 | 11/11/2023 | Ricardo Reyes Jr. | This is the first draft of the design template |

**Instructions**

The document will cover the design for the Draw It or Lose It game a game by The Game Room. My attempt with this document is to describe the conversion of the current Android App Game to Web Base App.

## [Executive Summary](#_sbfa50wo7nsh)

Changing the game from Android to web base will require several adjustments to the code also I will suggest using spring boot on the backend. This will allow us to use Java and not have to convert everything to a new language to be able to run the web app.

## Requirements

*I will suggest for the front end of this web app to build it with Next.js as a framework. Once again, I will suggest using Spring Boot for the back end and Java. For a database I would suggest using a non-relational cloud-based database this will allow flexibility and growth.*

## [Design Constraints](#_2et92p0)

There may be some issues with converting the Android app to

Web apps one the frontends of each apps are different. Developing a new front end for the web app also making sure that the screen sizing and resolutions have been considered.

## [System Architecture View](#_ilbxbyevv6b6)

Having the web base app server render will allow more users with lower spec computers to be able to use the app. Using a cloud base database will allow flexibility and growth. If the Android app has been developed with Java vs Kotlin will allow the conversion a lot easier being if it is Java with Spring Boot on the back end this will allow us to have a standalone web base application. As for the front end we’ll be using Next.js.

## [Domain Model](#_8h2ehzxfam4o)

<Describe the UML class diagram provided below. Explain how the classes relate to each other. Identify any object-oriented programming principles that are demonstrated in the diagram and how they are used to fulfill the software requirements efficiently.>

**"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)

Having a web application developed with these frameworks and languages will allow the web base app to run on all operating systems from Windows, MacOS, and Linux this application will run smoothly on browsers like Edge, Safari, Mozilla, Chrome, etc.

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | MacOS is a very simple to use OS on the client side. Yet when it comes to the development, you’ll run into issues with licensing. Mac has one of the most expensive licensing on the software. | Linux is free to use and for licensing it is the same. Linux being open source and with the community often looking out for each other the security is pretty high with Linux. | The fees are lower with Windows, but Windows securities are much more vulnerable than MacOS and Linux. | N/A |
| **Client Side** | Since the game will be running on the server. The one thing we’ll need to do is make sure that it will run on all modern browsers. For Mac it will mean making sure Safari is compatible. | Same as with Mac so long as the game is compatible with modern browsers. As for Linux we are talking about Mozilla, Chromium, as well as Chrome. | For Window is the same as with Linux and MacOS. We need to make sure the game is compatible with all modern browsers we’ll need to include Edge since this is the browser that ships out with Windows. | There are different ways we can use here. We could create a mobile app for each specific device. Such as using Kotlin or Java for Android or using Swift for iOS. We could have the entire webapp that has been developed for the browsers be responsive to be able to work on these devices. By going the first route required us to maintain only the webapp and not having to maintain the webapp, the Android and the iOS version. |
| **Development Tools** | We have the development tools that come with the browser having a dynamic webapp using Nextjs. | Same as with MacOS we have the development tools that come with every browser. Using frameworks will help us maintain the webapp easier. | With Windows as with Linux and MacOS the development tools and taking advantage of the frameworks to keep the WebApp dynamic. | For the mobile devices if we create individual apps for each mobile device one for Android and one for iOS. We would need to use the individual IDEs for each Xcode for iOS and IntelliJ for Kotlin. If we are to use the webapp there is not development tools other than the ones we used for development for the webapp. |

## 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 am recommending Linux openSUSE for Draw It or Lose it. It is stable and has a rolling release named Tumbleweed.
2. **Operating Systems Architectures**: Linux based systems are very stable and secure and compatible with other operating systems such as Windows and MacOS.
3. **Storage Management**: Having cloud storage along with git and GitHub to keep the project on a repository keeping the version control of the project and having the project safe as it is being developed. Use a non-relational database for the game and user data.
4. **Memory Management**: MacOS divides the available RAM into two broad sections. It reserves for itself a zone or partition of memory known as the system partition. The game will use the user’s ram and will have a memory watcher for the loads on the user’s system.
5. **Distributed Systems and Networks**: Since the game is on a cloud. The game does not have to stop. You can just move to another server in the cloud if that server crashes without affecting the users experience causing lag or loss of play time.
6. **Security**: I suggest using a 2-step verification to access the game. Have the different levels separated (e.g. admin, game, team, players, users, etc.)