

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template** 1](file:///C:\Users\amand\Downloads\CS%20230%20Project%20Software%20Design%20Template.docx#_Toc115077317)

[**Table of Contents 2**](file:///C:\Users\amand\Downloads\CS%20230%20Project%20Software%20Design%20Template.docx#_Toc115077318)

[**Document Revision History 2**](file:///C:\Users\amand\Downloads\CS%20230%20Project%20Software%20Design%20Template.docx#_Toc115077319)

[**Executive Summary 3**](file:///C:\Users\amand\Downloads\CS%20230%20Project%20Software%20Design%20Template.docx#_Toc115077320)

[**Requirements 3**](file:///C:\Users\amand\Downloads\CS%20230%20Project%20Software%20Design%20Template.docx#_Toc115077321)

[**Design Constraints 3**](file:///C:\Users\amand\Downloads\CS%20230%20Project%20Software%20Design%20Template.docx#_Toc115077322)

[**System Architecture View 3**](file:///C:\Users\amand\Downloads\CS%20230%20Project%20Software%20Design%20Template.docx#_Toc115077323)

[**Domain Model 3**](file:///C:\Users\amand\Downloads\CS%20230%20Project%20Software%20Design%20Template.docx#_Toc115077324)

[**Evaluation 4**](file:///C:\Users\amand\Downloads\CS%20230%20Project%20Software%20Design%20Template.docx#_Toc115077325)

[**Recommendations 5**](file:///C:\Users\amand\Downloads\CS%20230%20Project%20Software%20Design%20Template.docx#_Toc115077326)

## [Document Revision History](file:///C:\\Users\\amand\\Downloads\\CS%20230%20Project%20Software%20Design%20Template.docx" \l "_grjogdjh5fi8)

| Version | Date | Author | Comments | |
| --- | --- | --- | --- | --- |
| 1.0 | 03/20/24 | Amanda Nelson | | Taking an Android only game and making it web based to run on multiple platforms. |

## [Executive Summary](file:///C:\\Users\\amand\\Downloads\\CS%20230%20Project%20Software%20Design%20Template.docx" \l "_sbfa50wo7nsh)

The client, The Gaming Room, has asked Creative Technology Solutions (CTS) to facilitate the development of the web-based version of their Android gaming app, Draw It or Lose It. The game must be able to run on multiple platforms. The application will get images for the game from a large library of stock drawings.

## Requirements

* A game will have the ability to have one or more teams involved.
* Each team will have multiple players assigned to it.
* Game and team names must be unique to allow users to check whether a name is in use when choosing a team name.
* Only one instance of the game can exist in memory at any given time. This can be accomplished by creating unique identifiers for each instance of a game, team, or player.

## [Design Constraints](file:///C:\Users\amand\Downloads\CS%20230%20Project%20Software%20Design%20Template.docx#_2et92p0)

* Game will have one or more teams involved.
* Each team will have multiple players.
* Game and team names must be unique to allow users to check whether a name is in use.
* Only one instance of the game can exist in memory at any given time.
* Needs to be able to run on multiple platforms.

## [System Architecture View](file:///C:\Users\amand\Downloads\CS%20230%20Project%20Software%20Design%20Template.docx#_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](file:///C:\\Users\\amand\\Downloads\\CS%20230%20Project%20Software%20Design%20Template.docx" \l "_8h2ehzxfam4o)

Looking at the UML class diagram below we can see the Entity class is connected to the Game, Team. And Player classes. The purpose of the Entity class is to hold common attributes and behaviors. For example, the classes Game, Team, and Players all have the attributes of id and name, as well as the behavior (or accessors) of getId and getName. This means the UML diagram shows that the classes will inherit from Entity. This means that the Entity class is a superclass. The classes relate to each other by the 0…\* lines connecting them. This means that the GameService has a zero to many (0…\*) instance of Game. Game has a instance of zero to many teams, and teams has zero to many players.

**"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](file:///C:\\Users\\amand\\Downloads\\CS%20230%20Project%20Software%20Design%20Template.docx" \l "_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 has limited software support compared to others. Great choice for web application hosting. | Linux has a weakness for those not already familiar with it. Extensive server software supposer and great for web servers | Windows may not be as secure as others. Great for .NET applications and SQL server databases. | Mobile Devices need to run on multiple operating systems, and they will have limited resources compared to servers. |
| **Client Side** | Mac can be expensive for user. Most users say it is easy to learn and operate the system. | Linux is mostly free to use, some purchasing may be required. | Cost can range depending on services. Understanding of UI/UX design principles. | Development for mobile devices needs testing on various devices and OS versions. Also will require platform-specific expertise. |
| **Development Tools** | Common IDEs include Visual Studio Code and Postman. Programming languages include Ruby, Python, PHP, and Node.js. Among many others. | Common IDEs include Visual Studio, PyCharm, and Eclipse. The best one for Linux is Spacemacs. Programming languages for Linux include Java, C++, and Python. Among many others. | Common IDEs include Visual Studio, PyCharm, and Eclipse. Programming languages include C, and C++. Which can be easy to learn. Among many others. | Programming tools that are used to build this type of software for deploying on Mobile Devices are React Native, Android Studio, and NativeScript. Common languages used are Java and Python. Among many others. |

## Recommendations

1. **Operating Platform**: In order to expand Draw It or Lose It to other computing environments, it is my recommendation that a cloud approach is used. Such as Microsoft Azure. This is due to flexibility and being budget friendly for the client. Microsoft is also budget friendly and easy to use.
2. **Operating Systems Architectures**: Microsoft Azure includes data centers that have physical servers, complex network infrastructure, and virtualized hardware. Azure also uses Kubernetes and offers different tiers based on needs, such as the standard tier for large-scale applications.
3. **Storage Management**: Microsoft Azure offers cloud-based storage called Azure Storage. This will make storage easier for the client to get more storage when adding in new pictures for the game. This will ultimately save on
4. **Memory Management**: For the Draw It or Lose It software, Azure uses memory management techniques such as minimizing memory fragmentation to avoid high usage and server load. They also have Azure Monitor, this is to monitor memory usage, to predict if more resources are needed.
5. **Distributed Systems and Networks**: Azure has what is called Azure Service Fabric. This is a distributed systems platform. Its purpose is to facilitate the development, deployment, management, and scaling of an application that is designed to run across multiple platforms. Which is essentially what the client wants with the game.
6. **Security**: Security capabilities of the recommended operating platform include multiple firewalls, storage security, defender for the cloud, App service authentication / Authorization, and log monitoring to just name a few.

In conclusion, Microsoft Azure can be completely customized based on clients’ needs and wants.

**Resources**

https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-azure/?&ef\_id=\_k\_Cj0KCQjw8pKxBhD\_ARIsAPrG45lqHYmb3FkFW4z3EKW2d6DppC00cFsz\_6I0tfmwE9Ix\_yor76tcM88aAnigEALw\_wcB\_k\_&OCID=AIDcmm5edswduu\_SEM\_\_k\_Cj0KCQjw8pKxBhD\_ARIsAPrG45lqHYmb3FkFW4z3EKW2d6DppC00cFsz\_6I0tfmwE9Ix\_yor76tcM88aAnigEALw\_wcB\_k\_&gad\_source=1&gclid=Cj0KCQjw8pKxBhD\_ARIsAPrG45lqHYmb3FkFW4z3EKW2d6DppC00cFsz\_6I0tfmwE9Ix\_yor76tcM88aAnigEALw\_wcB