

The Gaming Room

# **CS 230 Project Software Design Template**

Version 3.0

## Table of Contents

[**CS 230 Project Software Design Template**](#_30j0zll)1

[**Table of Contents**](#_1fob9te)2

[**Document Revision History**](#_2et92p0)2

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

[**Design Constraints**](#_3dy6vkm)3

[**System Architecture View**](#_1t3h5sf)3

[**Domain Model**](#_4d34og8)3

[**Evaluation**](#_2s8eyo1)4

[**Recommendations**](#_3rdcrjn)5

## [Document Revision History](#_2et92p0)

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| 1.0 | 02/03/21 | Abigail Mattingly | This write up will discuss the gaming project. It will talk about what was done and what still needs to be done and any errors that were encountered. |
| 2.0 | 02/07/21 | Abigail  Mattingly | This write up will discuss advantages and disadvantages of different operating platforms in depth in reference to client and server side as well as developmental tools. |
| 3.0 | 02/21/21 | Abigail Mattingly | This write up will discuss the recommendations to various system architectures to be able to run the application smoothly. |

## [Executive Summary](#_tyjcwt)

Creative Technology Solutions has given the task of creating a web-based game “Draw it or Lose it”. The app is currently available on androids, but they would like it to be available on multiple platforms. I will first look over and take notes on the android app. I should then be able to complete this project by creating a milestone to be checked to first receive any feedback. After that is complete I will finish the remainder of the project. Please note that this may take time given that transforming a game only available on one operating system can be hard if the software that was used is not widely accepted by other operating systems. If more people are needed due to needing multiple teams to have the project working on multiple platforms then it may cost more to complete the project.

## [Design Constraints](#_3dy6vkm)

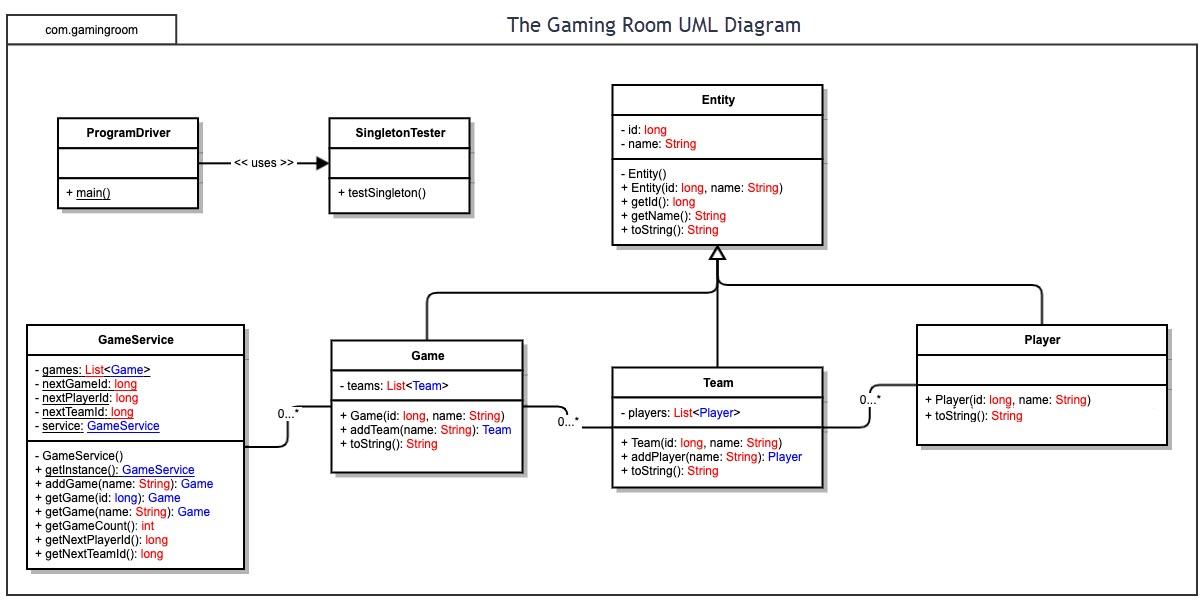
Design constraints in this project could be that the game is already developed on one operating system and switching it from an app to a web-based system can be challenging if the software that was used is not something that can be used for different operating systems. It also can be complicated to get a game to look the same across many platforms. If the design needs multiple different software that requires more teams to complete it then it may cost more. With the use of multiple software to be used it could also take longer to complete the project. My team and I will work closely to make sure proper precautions are taken and keep the client up to date with any new design constraints. We will also do our best to produce a product in a timely and costly manner.

## [System Architecture View](#_1t3h5sf)

“Draw it or Lose it” was built using seven classes in the language Java.

## [Domain Model](#_4d34og8)

The UML class diagram below demonstrates object-oriented principles. The classes Game, Team, and Player use inheritance because they are used in the Entity class. This helps other classes inherit attributes from classes that have relationships with the Entity class. Team and Player also have a relationship with each other. Another object-oriented principle that is used is encapsulation. The class Entity protects data that is used in the program by making sure public methods in the program can't access it which protects the player's information.

****

## [Evaluation](#_2s8eyo1)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| **Server Side** | Can be Expensive. User friendly graphical interface and workstation. Easy accessibility and configuration to the server. | Cost effective. Can be difficult to navigate the platform. Has a command shell for a simple configuration to server. | Can be costly. User friendly platforms and access to command prompt. Windows software is available. | Can be costly, but most users already have. Challenges in creating the game that can be compatible across most mobile platforms such as iPhone and android. |
| **Client Side** | Average time is needed to use. Adequate skill is needed. More expensive than Linux. | Maximum time is needed to use. Mastery of Linux is required. It cost minimum | Minimum time needed to use. Windows users provide easy use. More expensive than Linux. | Maximum time needed to use. Usually easy to use your own device. To support multiple operating systems can be challenging. |
| **Development Tools** | Most languages include JavaScript, CSS, HTML, Python, among many others. Tools include libraries, Eclipse, Visual Studios, and more. Most tools are free to use. To get advanced use on the applications it may cost more. | Most languages include Ruby on Rails, Java, Python, CSS, HTML, among others. Tools include nodejs, Visual Studios, Repl.it, command prompt, and Github. Most tools are free to use. To get advanced use on the applications it may cost more. | Most languages include Ruby on Rails, Java, Python, C++, C#, JavaScript, HTML, among others. Tools include Visual Studios, Repl.it, and command prompt. Most tools are free to use. To get advanced use on the applications it may cost more. | Most languages include Ruby on Rails, Java, Python, C++, C#, JavaScript, HTML, among others. Tools include Visual Studios, Repl.it, command prompt (power shell), and Github. Most tools are free to use. Some apps may require a small fee.To get advanced use on the applications it may cost more. |

## Recommendations

1. **Operating Platform**: There are many differences between games for mobile devices and portable computers. Through the use of the operator, event, game design, and software the games can be built very differently. All operating platforms have their advantages and disadvantages and compete to have the best quality platforms that a user would want. This helps the game developers make the best application for the client. I recommend windows for this because it is the most user friendly, has the most software, and widely accepted coding languages and developmental tools.
2. **Operating Systems Architectures**: A X86 with 32-bits would be my recommendation based on how simple it is to develop on this kind of platform. The wide range of hardware compatibility allows more software to work with it. This allows more operating systems to have universal games.
3. **Storage Management**: The biggest thing when thinking about storage for this application is that the client wants it to be available across many platforms. This in a way eliminates the use of a removable hard drive because not all operating platforms such as phones have easy access to that. By using the cloud for storing the data will always be accessible. Another advantage of the cloud is that it has a low risk of being destroyed. The application needs to be able to store how the game runs, the two hundred images that convert to 1600 megabytes, the user's information, and if the game needs to be saved. Users can also store multiple applications and personal files on the cloud as well. The cloud allows for additional storage to be purchased if room for storage becomes a problem.
4. **Memory Management**: When using windows, each process of a 32-bit Microsoft Windows has its own virtual address space. This allows addressing of up to 4 gigabytes of memory. Each process of a 64-bit Windows has a virtual address space of 8 terabytes. This means that all the threads can access their virtual address space, but not each other's space. This protects against corruption to any files.
5. **Distributed Systems and Networks**: With the use of multi-user interaction systems that are network-based such as games that typically include a database shared among the players would be the best. By allowing interaction with one another over the network it will better assist the running of the application. Developers have to implement the shared database and the communication between players from scratch.
6. **Security**: Nothing can be guaranteed completely safe. Everything has a risk. Data can accidentally be transmitted that is private information and can affect your personal information or technology. On the other hand, it is riskier not to have some sort of security that provides at least basic protection. When visiting risky sites such as games you can take protection matters such as limiting personal data. Developers are required to minimize data collection to assist in the privacy of the user. The best recommendation is to only provide what is needed.