

Draw It or Lose It

**CS 230 Project Software Design Template**

Version 1.0

**Table of Contents**

**Document Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| 1.0 | 01/23/2024 | Devin Criswell | First draft, fill in information |
| 1.1 | 02/4/2024 | Devin Criswell | Completed Evaluation section |
| 1.2 | 2/18/2024 | Devin Criswell | Completed Recommendation section |

**Instructions**

Fill in all bracketed information on page one (the cover page), in the Document Revision History table, and below each header. Under each header, remove the bracketed prompt and write your own paragraph response covering the indicated information.

**Executive Summary**

We will be creating a web-based version of Draw It or Lose It. For this application we will look to create a way to allow for multiple teams to play together. We will develop an interface that will allow multiple players on a team to interact with one another. We will require each team and game save to have a unique ID assigned to it.

**Requirements**

*This is to be a web based service.*

*Multiple teams and players on teams.*

*Unique names.*

*Only one game available at a time*

**Design Constraints**

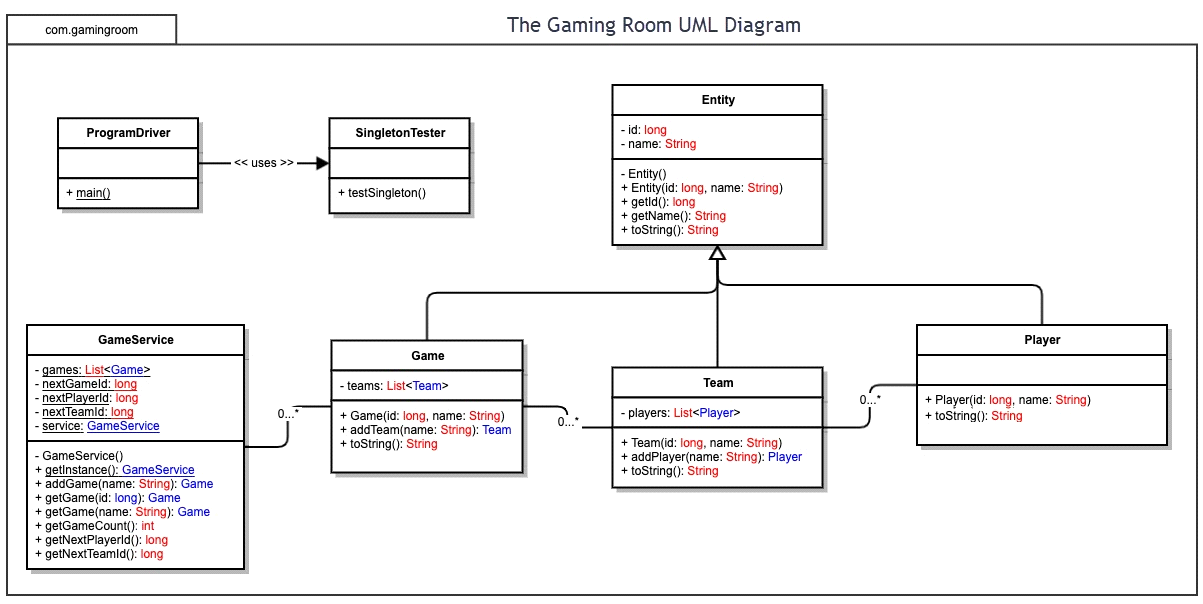
Some of the constraints we could face with a web-based game application are security, its reliability, and the scalability. We will need to provide players with the comfort and information about how we will protect their data and what exactly we will use their data for. We will need to establish secure servers that will be able to stand up to the demand of our application. With the desire of having multiple teams and players on teams we will need to ensure the application will run smoothly regardless of the amount of players using the application.

**System Architecture View**

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**

There is an Entity class that will serve as the base class. Entity will contain the id and name as private. There are three subclasses to Entity that are Game, Team, and Player. Game will extend the GameService class. Game, Team,and Player will contain the unique id's and names required for the application. ProgramDriver will contain the main().



**Evaluation**

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 can provide users with their own environment and infrastructure resources. Typically, Mac offers a more robust security. Mac is known for its quality build which could lead to a greater reliability. Limited to Apple hardware. | Linux is free and open source. This would mean cost savings and frequent updates. Linux is compatible with most hardware. Linux is harder to learn. Support for Linux can be harder to acquire. | Windows is the most widely used. Windows is more user friendly and provides an easier experience. Windows is compatible with all hardware. Windows does have security concerns for its users. | Mobile Devices can offer greater accessibility while allowing multiple different platforms. They tend to require less frequent updates. They require a constant wireless connection. They can be slower than other options. |
| **Client Side** | There needs to be an understanding of the Mac ecosystem. There are a variety of supported browsers. There is the ability for cross browser testing. Development on Mac can be more costly than other options. | Has a large variety of supported browsers. Its open source allows for it to be more compatible with more software. Can be iterated and updated quickly. This is typically the most cost-effective Operating system to develop with. | Can be tested easily between platforms. Has a large variety of supported browsers. It can be difficult to test on Mac and Apple products. Windows does support two architectures, 32 bit and 64 bit, that could add to development time. | There needs to be an understanding of mobile application development. It can be difficult to test on other browsers. Mobile can be a longer development. It can cost more to develop for mobile because of the range of platforms available. Mobile requires a different UX design. |
| **Development Tools** | There is the Apple Dev program for a yearly subscription. All applications must be approved by Apple. It can run programs using a virtual machine. Xcode is the official IDE for Mac and other OS products. The primary language is Swift. | Can run programs using a virtual machine. It does not have a subscription service cost. It can run multiple IDEs like VS code, CLion, or PyCharm. C and C++ are the popular languages used. It does not need to be approved before submitting. | There is no service cost for development tools. Visual Studio and VS code are the main IDEs used with Visual Studio being the official IDE for Microsoft. VS code has the ability to work with multiple programming languages. Additional libraries may come at a cost. It can use a virtual machine as well. | For developing for Apple a yearly Apple Dev subscription and approval for applications is required. Xcode is the IDE when working with iOS. Android Studio is Google's official IDE that is used for working with Android. Android Studio is free to use and download. Swift and Java are the primary languages. |

**Recommendations**

Analyze the characteristics of and techniques specific to various systems architectures and make a recommendation to The Gaming Room. Specifically, address the following:

* **Operating Platform**: For Draw It or Lose It, we recommend using Windows. Windows has the highest user base. It is also the most user friendly of the options. There is no upfront cost for developer tools and VS code has a wide range of compatible languages for developing with. The ability to expand is available for the growth of the application.
* **Operating Systems Architectures**: The operating system for Windows includes HAL, Driver, Microkernel, and Executive services. The architecture of Windows consists of two main components, the user mode and Kernel mode. Windows structure provides a high-level interface for various system features. The Windows system is typically powered by x86 / x64 processors but also can use Arm processors.
* **Storage Management**: We will utilize cloud-based storage for Draw It or Lose It. Since the game is to function online with multiple players and teams, cloud-based storage would best meet the needs for the game. Cloud based storage will also give us the ability to expand in the future as the game grows. Cloud based storage would also help to take away extra tasks by not having to manage the system because of the third party hosting the game.
* **Memory Management**: Windows uses memory management that tracks memory locations, allocates RAM, and manages the process. When a program is opened, the memory management system allocates RAM for it and tells the system when the program has been closed, freeing up memory. Windows task manager uses four different types of memory for executables, Commit, working set, shareable, and private.
* **Distributed Systems and Networks**: Since Draw It or Lose It will be a web-based application we will utilize a browser available on all platforms. Google's Chrome browser is available for download on all current platforms. The use of the browser would enable the game to reach the highest number of users without excluding others. This would also allow us to use Google's Cloud services for storage. Chrome also implements multiple means of security for its users.
* **Security**: Our application will go through different stages of testing. We will test for exploits between platforms and for specific platforms. With Google as a main piece in our system we will utilize their built-in security. Google regularly tests their systems and keeps billions of user’s data secure on a constant basis. They use multiple techniques to do this like predictive phishing protections, sandboxing, and site isolation to ensure users and their data are secure. Each user will have unique log in information that will be tied to their IP address. With these measures we believe we can provide our users with the comfort of knowing their information is secure.