Draw It or Lose It

CS 230 Project Software Design

Version 1.0

Table of Contents

Document Revision History

Version	Date	Author	Comments
1.0	11/13/2021	Kennedy Uzoho	Filled out the Evaluation summary table

Executive Summary

In this project, the Gaming Room wants a web-based gaming application named Draw It or Lose It. In the game, teams will compete to guess what is being drawn. Instead of a player drawing the images to help other team players guess image puzzles, the software application itself will render ready-to-draw images from a library of image drawings. The drawings are rendered at a rate that is steady and the drawing is completed in 30 seconds marked time. The game operates on Android devices, so the application needs to be rebuilt as a web-based application so users can run it on their respective OS.

Design Constraints

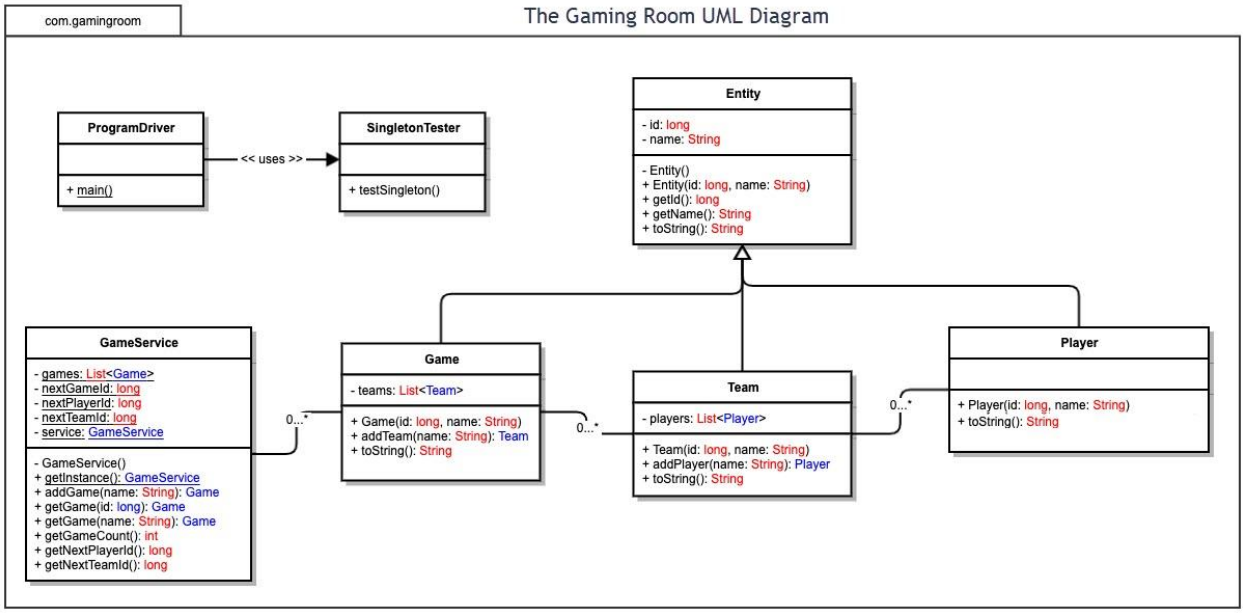
The major design constraints are that the application is written in Java, and is a web-based, online instance of the game that can exist in memory at any given time. There should be a way to make the application an online multiplayer game that runs on the web. Images sourcing and the first team has 30 seconds to guess correctly. Designing the software to be web-based, so the users could just log onto the website and play the game.

System Architecture View

Normally, in these sections, we should be describing the system and subsystem architecture present in the application, including physical components or tiers, but in this case it's N/A

Domain Model

Looking at the UML diagram, the three classes Game, Team, and Player class have a relationship with Entity. Game, Team, and Player classes are inherited from Entity, and Game, Team, and Entity classes all have common attributes which are the 'id' and 'name'. Between Team and Player, we can see that Team has a reference to Player and the same way Game has a reference to Team and GameService as the organizer has Game. Overall, the program will have multiple teams with multiple players in them. The teams will play the game against each other according to the game rules. The software will render a picture and one player will be drawing the picture and the other player will need to guess what the picture is within 30 seconds to win and get a point, otherwise, they will lose after 30 seconds marked time.



Evaluation

Development Requirements	Mac	Linux	Windows	Mobile Devices
Server Side	The terminal is flexible and there are commands to configure servers, however, it is expensive.	Open source, low cost-effective. Servers are more secure. However, it may be difficult to use on-premises to support web hosting, so other web-hosting agents may be required.	Windows are more available OS and has a command line (cmd), however expensive just like Mac OS	It is possible to use a mobile server, however, it may be very selective to smart mobile devices.
Client Side	Requires moderate expertise to operate. Similar cost to windows	It will require learning; however, the cost is cheap.	Requires minimum expertise, but the cost is similar to Mac	Mobile devices provide flexibility to clients and developers as well. However, this is trickier to implement compared to other devices.
Development Tools	Eclipse, GitHub, Notepad++, databases. With Swift you can mix in tools like Notepad and Mac c+an run all languages.	Linux supports Visual Studio, Eclipse, and Note pad. Linux is really about command prompts and servers.	Windows is easier to operate on but can run the same tools as Linux	Numerous apps can be created using Android and Swift. However, languages and apps can run on all of these machines.

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**: The platform that will be suitable for the gaming room to expand the software application for draw it or lose it would be the Windows platform because Windows OS is more available and cheaper for a starting point.

2. **Operating Systems Architectures**: The Windows operating system is essentially built with two distinct layers. One is the Graphical User Interface (GUI) that most users interact with, while the other is the kernel that operates with command-line inputs. This makes it easier for both developers and non-developers to use. Windows is well known for its popularity and versatility, especially regarding multi-programming options for developers.

3. **Storage Management**: Windows 10 and 11 are equipped with storage sense, this program lets the operator manage files on the hard drive and at the same time be able to see how much space files have taken up. The operator can choose where to save applications, easily create files, and move them around easily.

4. **Memory Management**: As the project is developed, there may arise a need to build a library containing images and other application files. Windows memory management is capable of handling up to 4GB memory space which can ensure smooth functioning of the applications.

5. **Distributed Systems and Networks**: In this case, each operating system runs very differently, so I did a little research and was able to come up with an idea that will require the company to make sure that the operating servers are strong and can support large volumes with other servers ready to take in case the current server crashes. There is the utility I researched is called Game Room, this utility can help change servers when the current one crashes.

6. **Security**: Windows has a built-in security system; however, for the users' data and information security it will be recommended to try another source equipped with encryption for additional security. The Windows remote/work environment comes with encrypted ESP, and this is functional when the users connect to the networks.