Kayla Macklin

Pt. 1: 05/21/25; Pt. 2: 06/05/25; Pt. 3: 06/17/25

Draw It or Lose It

CS 230 Project Software Design Template

Version 1.0

Kayla Macklin

Table of Contents

Document Revision History

Version	Date	Author	Comments
1.0	05/20/2025	Kayla Macklin	Updates were applied to several sections including the cover page, revision history, executive summary, requirements, design constraints, system architecture view, domain model, and evaluation and recommendations.
2.0	06/05/2025	Kayla Macklin	Updates were applied to several sections including the revision history, evaluation and recommendations.
3.0	06/17/2025	Kayla Macklin	Updates were applied to the architecture recommendation

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

The Gaming Room requested CTS to develop a web version of their Android game Draw It or Lose It. The game will feature multiple teams, each consisting of several players. Each player, team, and game instance will be unique. To prevent the creation of duplicate game instances, the design has been modified to use a singleton pattern. This change will help avoid conflicts between teams and their members.

Requirements

< Please note: While this section is not being assessed, it will support your outline of the design constraints below. *In your summary, identify each of the client's business and technical requirements in a clear and concise manner.>*

Design Constraints

The Draw It or Lose It game can now be played on Android devices in the Gaming Room. Java was selected as the technology for web deployment since CTS has been requested to make it available online. Using Java, the native language for Android SDK, should simplify this new process of launching it on the web.

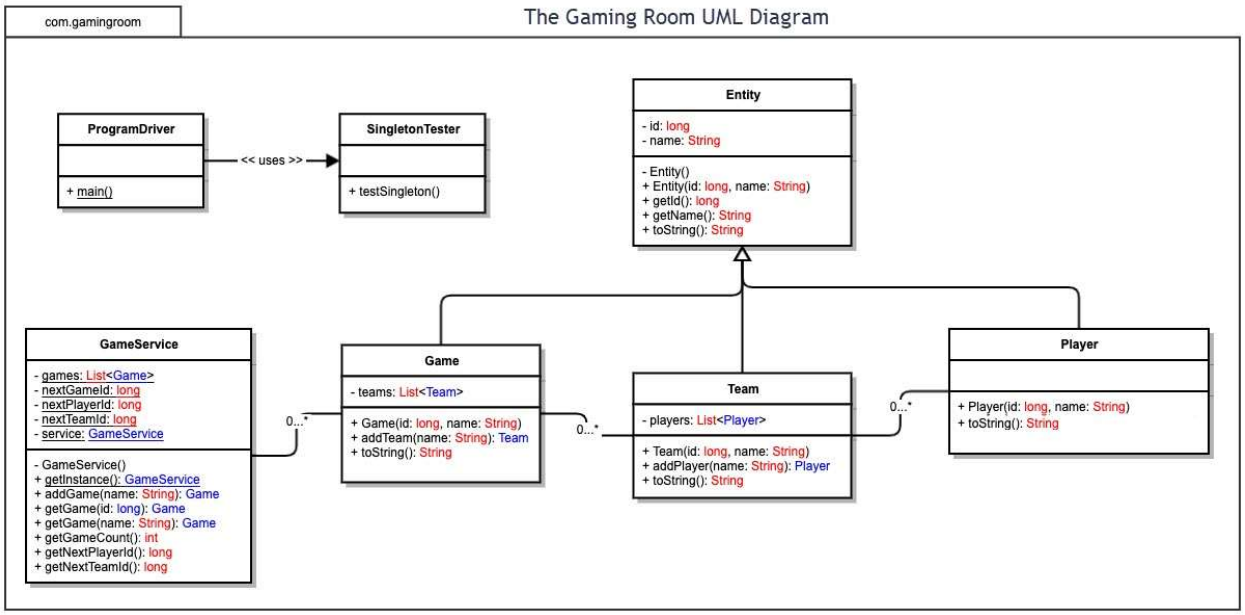
System Architecture View

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

The Entity connects the Game, Team, and Player classes, meaning they all get their information from Entity. In UML, we can represent this as inheritance, making Entity the main class. If we examine their connections, we find that Team and Player have a 'has a' relationship, while Game has a Team and Game Service has Games. In UML terms, this is called aggregation (HAS-A). When we say a user 'has a,' we mean it's an instance of one class that refers to another class's instance. Looking at the diagram, we can see that Game Service refers to Games, Games refer to Teams, and Team refers to Players.

Kayla Macklin



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 is user friendly and allows for easy server setup. It has a simple graphical interface that's easy to navigate, along with flexible terminal commands.	Affordable option. The platform is hard to use. Command shell makes it easy to set up and access the server.	The server side costs a lot. It has an easy-to-use graphical interface and includes a command prompt.	Other devices have better specs. Mobile device specifications differ from one user to another.

Client Side	Costly for users. Requires a fair amount of time and knowledge. Accurate skills are necessary to use the operating system.	Using Linux takes a lot of skill and time. You need Linux data to operate the system, and it can be pretty expensive for users.	Costlier than Linux systems. Simple to learn and grasp how to manage a Windows setup. Requires minimal expertise.	Allow clients and developers to access updates from anywhere, though it might be a bit harder to set up compared to other devices.
Development Tools	Languages like HTML, CSS, and JavaScript are essential for frontend development, along with libraries and tools such as PyCharm, GitHub, and Visual Studio.	Languages like HTML, CSS, and JavaScript are used for web development. There are libraries that help with frontend development and other programming languages. Linux systems also support languages like JavaScript, Ruby, PHP, and Python.	Languages like HTML, CSS, and JavaScript are used for web development. There are libraries that help with frontend development and various programming languages. Some popular developer tools are Eclipse, command prompt, and PyCharm.	Languages like HTML, CSS, and JavaScript are used for web development. There are libraries that help with frontend development, and IDEs for programming languages include HTML, PHP, C++, and Python.

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 recommend using the Windows operating system to expand the size of the game app. Right now, Windows is the most popular OS, with 45.3% of users choosing it. Since the app is web-based, it's logical to use Windows for the backend tasks like answering user questions and fixing issues. The platform offers more flexibility, but keep in mind that there are licensing fees involved.

2. **Operating Systems Architectures**: The Windows operating system features an easy-to-use graphical user interface (GUI) and offers a wide range of integrated development environments (IDEs) for app development. Some of the most popular IDEs include Visual Studio, Eclipse, IntelliJ, JetBrains, and others.

3. **Storage Management**: People who handle the back end can easily figure out the settings since Windows OS makes storage management pretty straightforward. Getting a Windows cloud server might cost extra, but it's a smart choice because clients can increase their storage if they need to add more stuff, expand their game app, and so on.

4. **Memory Management**: Windows OS users have two choices: virtual memory and physical memory. Virtual memory is better because it helps run big programs smoothly. Plus, it offers important advantages like memory protection and longer usage compared to physical memory.

5. **Distributed Systems and Networks**: Using Windows OS might lead to some issues. Team members should talk to each other to solve common problems like lagging, repeated queuing, and overloaded servers that can make even easy tasks tough.

6. **Security**: A lot of safety measures are taken into account. This includes the Windows Defender antivirus software, which is always running on the user's computer. VPN services are also part of this to better protect user information. Because of this, it's really important to do daily security checks to make sure users are safe and to stop anyone from getting into their personal data. This also means making sure that the people who create the app are well-trained in keeping user information safe and using encryption.