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	04/23/2025	Najah Thompson	<Brief description of changes in this revision>

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 has requested the development of a web-based version of their Android game, “Draw It or Lose It," where teams compete to guess drawings. The game must support multiple teams and players, with unique identifiers for each game, team, and player to avoid conflicts. Only one instance of the game should run at a :me, ensuring no duplication. Our solution will focus on using unique identifiers and a scalable database to manage game data and user interactions. The next steps will involve finalizing the system architecture and beginning the development of core features to ensure a seamless, multi-platform user experience.

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 :me. This can be accomplished by creating unique identifiers for each instance of a game, team, or player.

Design Constraints

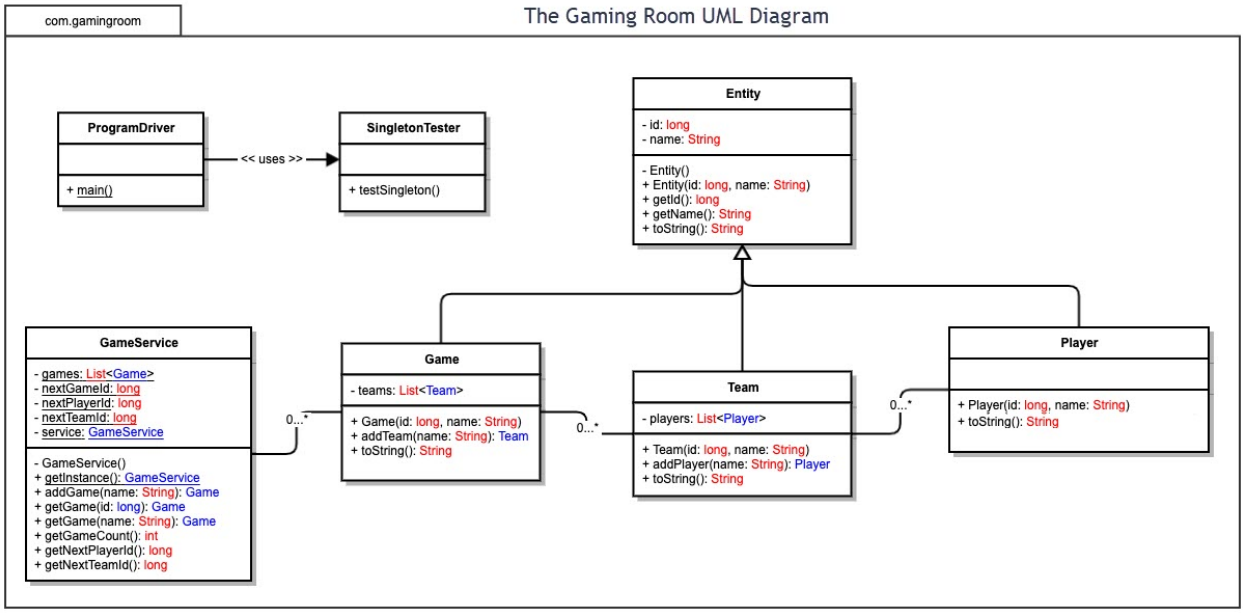
In a web-based distributed environment, the game application must be designed to handle scalability, ensuring it can support a large number of users and real-:me interactions. Unique identifiers for games, teams, and players are crucial to prevent data conflicts and ensure proper game management. The application must be cross-platform compatible, providing a seamless experience across various browsers and devices.

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

As we follow the diagram The first class we see is the ‘ProgramDriver’ class which is the entry point to the application. It interacts with the ‘SIngletonTester’ Class and is responsible for controlling how the game flows and initiating the game.
The ‘SingletonTester ‘Class ensures that a class has only one game at a :me throughout the application.
The ‘Entity ‘Class serves as a base class inherited by the Game , Team and Player classes.
GameService Class is connected to the Game class , managing multiple game objects. It is the container for Game objects.
The Game lass has a relationship with Team. Showing that a single game can involve multiple teams.
Team is linked to player class and consist of multiple player objects.



Evaluation

Because Linux is a popular, dependable, and cost-free server operating system, it is the ideal choice for this project's web-based application. Windows is also usable, albeit there can be license fees. Servers don't typically run macOS. Since the application will operate on a web browser on the client side, it is easier to support Windows, Mac, Linux, and mobile (iOS and Android) devices by utilizing contemporary web technologies like HTML5, CSS, and JavaScript. React and other cross-platform solutions can save time and eliminate the need for separate teams. The majority of programming tools, such as Git and Visual Studio Code, are free and compatible with all platforms.

Development Requirements	Mac	Linux	Windows	Mobile Devices
Server Side	Supports web-based applications but they aren't as common for product servers,.	This is the most ideal choice , its free and secure making it the most valuable option.	Work well with IIS and Visual Studio , licensing cost may be an issue	No server hosting on the mobile devices. Most phones do connect to the web app browsers
Client Side	Comopatible with all browsers , can make use the app is responsive	Fully supports browsers based web apps.	Supports all browsers	Most can run responsively with web apps etc
Development Tools	Xcode and terminal tools are used to support the wen and IOS devices. There aren't any extra cost for the tools.	Supports Open SOurce tools.	Can use visual studio and other tools . Some may require licensing	Responsive with frameworks to help build for mobile

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:**
   A cloud-based platform is the best choice for this application. It supports scalability, high availability, and makes it easier to deploy across multiple devices and operating systems.

2. **Operating System Architectures:**
   A microservices architecture is recommended. This breaks the application into smaller, independent services that each handle specific tasks. It allows for easier updates, better performance, and faster recovery if something goes wrong.

3. **Storage Management:**
   A cloud-hosted database will be used, with features like replication and automated backups to ensure data is always available and secure. This setup supports fast access and protects against data loss.

4. **Memory Management:**
   Using a cloud-based virtual environment, memory and other resources (like CPU) will be dynamically allocated based on demand. Tools like Redis or Memcached can be used to cache active game sessions and frequently accessed data to improve performance. Elastic scaling ensures the system adjusts automatically based on how many people are playing.

5. **Distributed Systems and Networks:**
   "Draw It or Lose It" will use a distributed architecture, meaning parts of the game (like the front end and back end) can run separately but communicate in real time over the internet. This supports smooth gameplay even if traffic increases or if a part of the system goes down. If a player loses connection, the game will save their progress so they can continue when back online.

6. **Security:**
   Security features will include network protection, user authentication, and data encryption to keep user information safe across all platforms. These measures help build trust and protect against breaches