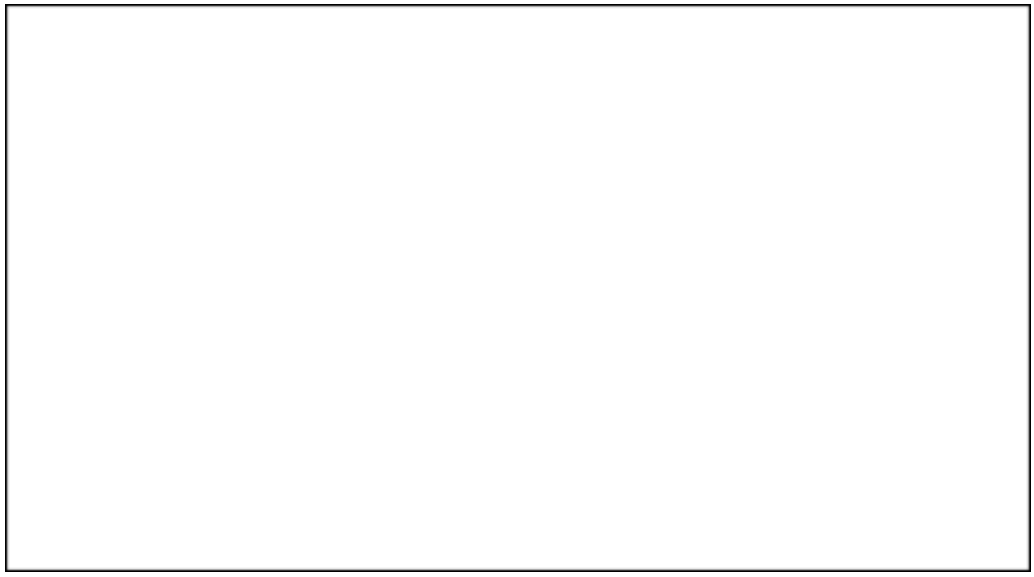


<Name-of-Software-Application>
CS 230 Project Software Design Template
Version 1.0

Table of Contents

Document Revision History

Version	Date	Author	Comments
1.0	5/20/2025	Kasra Pratt	+ Executive Summary + Design Constraints
1.1	6/2/2025	Kasra Pratt	+ Evaluation

2.0	6/20/2025	Kasra Pratt	+ Recommendations

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

<Write a summary to introduce the software design problem and present a solution. Be sure to provide the client with any critical information they must know in order to proceed with the process you are proposing.>

The Gaming Room wants to revamp their game, Draw It or Lose It, from an Android-only application to a web-based application accessible across multiple platforms. The solutions contained within this document leverage object-oriented principles, a singleton service for game management, and robust class features for a robust, scalable, and maintainable web-based game.

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

<Identify the design constraints for developing the game application in a web-based distributed environment and explain the implications of the design constraints on application development.>

Developing Draw It or Lose It as a web-based application in a distributed environment introduces a few levels of complexity. The application must be compatible with various browsers and devices, which requires a well-developed front end and a robust backend server. Multiple game sessions may run simultaneously, which requires efficient session management and data concurrency across distributed clients. Managing timers, rendering drawings, and handling concurrent sessions are resource-intensive tasks that require efficient resource handling. These constraints suggest a modular architecture with a centralized service to manage game instances and an optimized front end for low latency.

System Architecture View

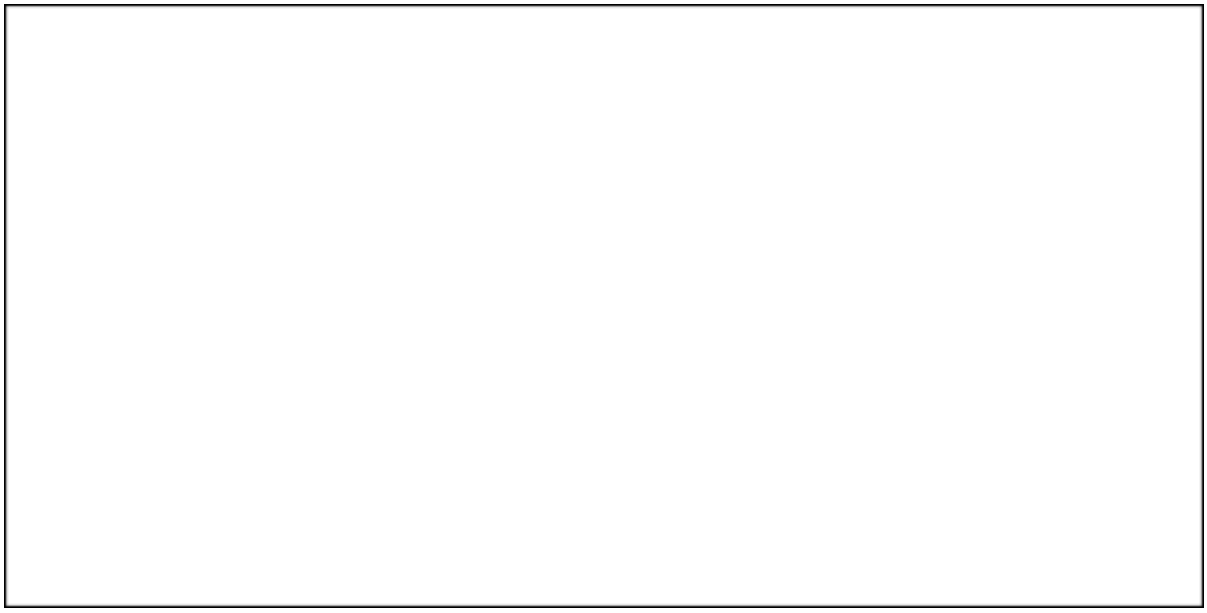
Please note: There is nothing required here for these projects, but this section serves as a reminder that describing the system and subsystem architecture present in the application, including physical components or tiers, may be required for other projects. A logical topology of the communication and storage aspects is also necessary to understand the overall architecture and should be provided.

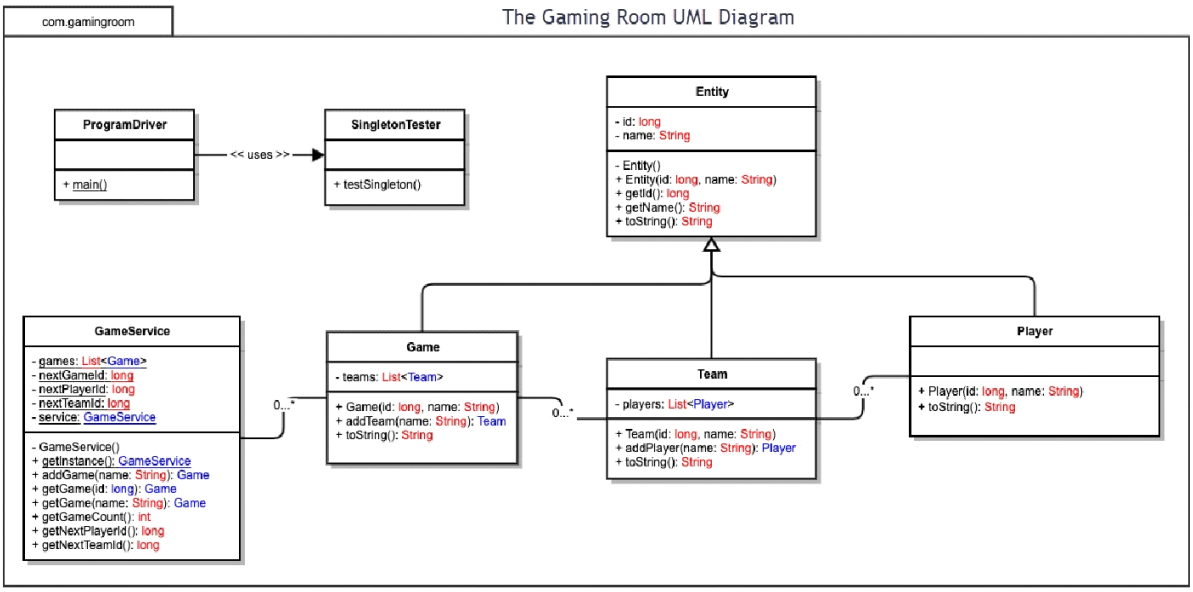
Domain Model

<Describe the UML class diagram provided below. Explain how the classes relate to each other. Identify any object-oriented programming principles that are demonstrated in the diagram and how they are used to fulfill the software requirements efficiently.>

The UML Class diagram below shows the following classes, their relationships, and functions.

- Entity - A base class with attributes Id (long) and name (String), providing common properties for all entities within the game. It uses inheritance to share these attributes with derived classes.
- Game - Game extends the Entity class, representing a game session. It contains a list of teams and manages multiple teams within a given instance.
- Team - Teams extend Entities, and represent a team within a game. It contains a list of players associating multiple players with a given team.
- Player - Players extend Entities, which represent players within a given team.
- GameService - GameService is a standalone class that manages all Game instances. And holds a single-to-many relationship with the aforementioned classes.





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, although secure and developer-friendly, is not suitable for web hosting as the cost-to-performance ratio and lack of customization make it better suited as a testing server.	Linux, currently running 90% of the internet infrastructure, excels at hosting web-based applications due to its open-source nature and high customizability.	Windows, although it boasts extensive support for Microsoft technologies and technical assistance, lags behind Linux due to efficiency and effectiveness, making it a situational choice.	Mobile devices are entirely impractical for hosting web applications due to many factors, including but not limited to cost-to-performance, customizability, lack of processing power, memory, and storage.
Client Side	Mac, Windows, and Linux will all have the same client-side code, given that we are operating using web browsers.	Mac, Windows, and Linux will all have the same client-side code, given that we are operating using web browsers.	Mac, Windows, and Linux will all have the same client-side code, given that we are operating using web browsers.	Mobile devices, such as smartphones, will require more development to create a user-friendly UI
Development Tools	Deploying web applications on Mac usually entails using Python, Ruby, or JavaScript, with Xcode serving as the primary development environment, which provides a comprehensive suite for mobile IOS development.	Linux supports a wide range of programming languages and popular IDEs, including nvim, vscode, and eclipse, while also supporting tools such as Apache and Nginx.	Windows development excels with programming languages like C#, C++, Visual Basic, ASP.NET, and Visual Studio as the development environment. Licencing cost is a factor to consider for Windows servers.	Mobile development uses platform specific tools (e.g. swift, kotlin) and IDEs (e.g. Xcode, Android Studio). Given the project circumstances, google chromes development tools will help style for mobile devices.

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**: <Recommend an appropriate operating platform that will allow The Gaming Room to expand Draw It or Lose It to other computing environments.>

- **Operating Systems Architectures**: I recommend Linux as the operating platform to support the expansion of Draw it or Lose It. Linux is an ideal choice as it is open-source, has no licensing costs, and is extensively customizable.

- **Storage Management**: I recommend using a PostgreSQL or equivalent Database to store information such as users, roles, permissions, game history, and images. Additionally, I suggest utilizing a RAID 5 array and ZFS to store all server and database data.

- **Memory Management**: Linux employs virtual memory management, which optimizes memory usage by allowing the system to treat disk space as an extension of RAM. This enables the server to handle multiple game instances and store more data than what physical memory allows.

- **Distributed Systems and Networks**: Draw it or Lose it will leverage a distributed system architecture using HTTPS protocols for standard client-server interactions and WebSockets for real-time, bidirectional communication. A centralized database will store game states, user data, session information, and more, ensuring consistency across clients. Risks such as outages are mitigated using load balancing and redundancy, allowing the system to reroute traffic in case of server failure, ensuring seamless communication across browsers on all devices.

- **Security**: Security is paramount, and Linux offers robust tooling to meet our needs. Linux provides features like firewalls to control network traffic, SELinux for access control, and frequent updates for security vulnerabilities. Data communications will be encrypted using HTTPS and SSL/TLS certs. Users will be authenticated using OAuth. Validations for SQL injections and XSS vulnerabilities will be implemented, and browsers will use recur content policies to enhance security. All employees and personnel should be regularly trained and informed to prevent social engineering vulnerabilities.