THE GAMING ROOM

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	09/25/2023	Caio Mauro	OS Information added
2.0	10/9/2023	Caio Mauro	Completed final 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

The software design problem involves the development of a web-based distributed game application called "Draw It or Lose It" for our client, The Gaming Room. This application is intended to provide an interactive and enjoyable gaming experience for users. The solution proposed involves the creation of a robust and efficient software architecture that adheres to the client's business and technical requirements. It includes a well-defined domain model, leveraging object-oriented programming principles, and addresses various design constraints associated with web-based environments.

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

Designing the "Draw It or Lose It" game in a web-based distributed environment introduces several constraints and possible implications:

1. Web-Based: The application must be accessible via web browsers, requiring compatibility with popular web technologies like HTML, CSS, and JavaScript.
2. Scalability: To accommodate a potentially large user base, the app should be scalable and capable of handling large user loads and lots of data.
3. Cross-Browser Compatibility: The application must function properly across different web browsers. This calls for lots of testing and extensive tweaking based on each browser and what they support.
4. Data Security: Given the sensitive data (e.g., accounts, scores), robust security measures such as encryption and authentication are essential to protect user information.
5. Latency: In a distributed environment, network latency can affect real-time interactions. Minimizing latency is crucial for smooth gameplay and an enjoyable user experience.
6. Platform Diversity: Supporting multiple platforms (Mac, Linux, Windows, and mobile devices) requires considering the unique tools and development requirements needs for each.
7. Resource Management: Efficient management of server resources, including memory and storage, is essential for a responsive and cost-effective application.

System Architecture View

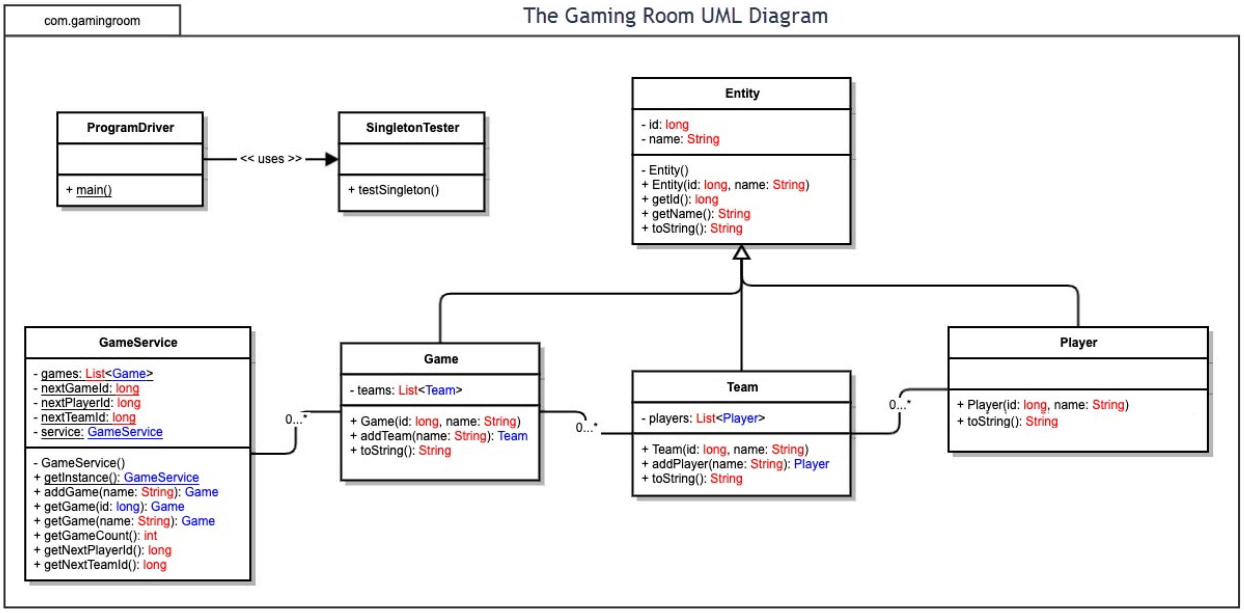
Please note: There is nothing required here for these projects, but this section serves as a reminder that describing the system and subsystem architecture present in the application, including physical components or tiers, may be required for other projects. A logical topology of the communication and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

Domain Model

The UML class diagram outlines the Domain Model for the "Draw It or Lose It" game. It consists of the following key classes:

- ProgramDriver: The entry point of the application.
- SingletonTester: Responsible for testing the singleton design pattern.
- Entity: A base class holding common attributes (id, name) and behaviors.
- GameService: Manages game instances, teams, players, and game-related operations.
- Game: Represents individual game sessions, each consisting of multiple teams.
- Team: Contains players and manages team-related operations.
- Player: Represents individual users participating in the game.

The relationships between these classes demonstrate object-oriented programming principles, including inheritance (Entity as a base class), association (GameService connected to Game, Game connected to Team, and Team connected to Player), and the use of singleton design pattern (SingletonTester). This model is in line with the software requirements by organizing game-related entities and their interactions in a structured manner. This facilitates the development of "Draw It or Lose It".



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	**Characteristics:** Mac offers a stable and reliable server environment. **Advantages:** It's Unix based, which is well-suited for web hosting. **Weaknesses:** Limited server software options compared to Linux which is more common.	**Characteristics:** Linux is a powerful and efficient server platform. **Advantages:** It offers a wide range of server software and is cheap. **Weaknesses:** Requires a lot more expertise to setup.	**Characteristics:** Windows Server is use friendly and supports various web hosting tools. **Advantages:** Easy to navigate interface for administrators. **Weaknesses:** Most likely has higher licensing costs.	**Characteristics:** Mobile can access web apps, but hosting would require cloud services. **Advantages:** Highly scalable through cloud hosting. **Weaknesses:** Limited control over the server hosting environment.
Client Side	**Considerations:** Developing for Mac requires knowledge in web technologies (HTML, CSS, JavaScript) and responsive design. Cost and time are relative. Requires extra testing on Safari.	**Considerations:** Linux client-side development involves similar web technologies as Mac. Cost and time are also relative. Compatibility testing is needed across Linux distributions.	**Considerations:** Windows client-side development follows standard web practices. Cost and time are low. Testing across different Windows versions and browsers is very important.	**Considerations:** Requires specific knowledge in mobile app development (iOS and Android) and responsive web design for browsers. Costs and time can be higher due to each platforms needs
Development Tools	**Languages and Tools:** Web technologies (HTML, CSS, JavaScript). IDEs like Visual Studio Code are commonly used. Licensing costs for development tools are minimal.	**Languages and Tools:** Similar to Mac, Linux development uses web technologies, with open-source IDEs like VSCode or JetBrains. Licensing costs are low.	**Languages and Tools:** Windows development also uses web technologies. IDEs like Visual Studio are popular. Licensing costs can be significant if using certain tools.	**Languages and Tools:** Mobile app development can require Swift (iOS) and Java/Kotlin (Android). IDEs like Xcode and Android Studio are used. Licensing costs depend on the platform and tools.

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:**
   Out of the most popular operating systems, Windows makes our final recommendation. It is flexible to help expand Draw It or Lose It to other systems in the future. We recommend Windows because:
   - Has support for Android Studio to expand into the mobile world. There are many frameworks like React Native which will help with this process.
   - Microsoft has built an extensive suite of products made for developers to design and deploy their products. Combined with the largest "gamer" userbase there are many options for software to aid in the development process for windows and other platforms.
2. **Operating Systems Architectures:**
   Windows has two modes: user and kernel. User mode functions are client-facing and affect most of what the client interacts with. Kernel mode is very technical, dealing with memory management, networking, and interacting directly with hardware. Windows saves data in a simple directory structure which is very easy to navigate.
3. **Storage Management:**
   Microsoft Azure takes our recommendation for storage due to its high level of support and easy integration with the Windows platform. Azure has more features as well and some of them are:
   - Cloud based storage allows for easy scaling.
   - Cost efficient and easy to manage with lots of up time.
   - Support for the deployment of different development environments like Docker.
4. **Memory Management**:
   Windows 11 is the newest version of the OS, and it has seen great improvements in all-around performance including memory management. It has refined algorithms to allocate and deallocate memory more efficiently which reduces memory leaks and slowdowns. Windows 11 optimizes memory utilization by better managing application windows, ensuring smoother multitasking.
5. **Distributed Systems and Networks:**
   Azure provides a lot of support when it comes to distributed systems and Networks:
   - Azure provides consistent uptime plenty of monitory solutions to maintain consistent gameplay for users. Azure makes it easy to tell when you need to scale up.
6. **Security:**
   Windows and Azure are two very secure tools. With some diligent security protocols, the protection of user date and private information will be well guarded. Azure has many monitory tools for security as well as well as built in features for the task:
   - VPN use is supported for enhanced security.
   - Passwords, 2fa, and SSL certificates to protect important data.
   - Encryption and Obfuscation is supported by both Windows 11 and Azure to add another layer of security.