

Draw It or Lose It

**CS 230 Project Software Design Template**

Version 3.0

**Table of Contents**

CS 230 Project Software Design Template\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_1

Table of Contents\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2

Document Revision History\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2

Executive Summary\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3

Requirements\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3

Design Constraints\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3

System Architecture View\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3

Domain Model\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3

Evaluation\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4

Recommendations\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**5**

**Document Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| 3.0 | 04/15/23 | Marc Aradillas | Client Recommendations |

**Executive Summary**

The Gaming Room is a company that wants to develop a web-based distributed game application called Draw It or Lose It. The game is intended to be played on different computing environments and mobile devices. The purpose of this software design document is to provide a clear understanding of the requirements, design constraints, and recommendations for developing the game application. The document includes an explanation of the UML class diagram, which demonstrates how the classes relate to each other and how object-oriented programming principles are used to fulfill the software requirements efficiently. The document also evaluates different operating platforms, operating system architectures, storage management, memory management, distributed systems and networks, and security. Based on the evaluation, the document recommends the appropriate operating platform, operating system architectures, storage management system, memory management techniques, and security measures to be used for the Draw It or Lose It game application.

**Requirements**

The client's business and technical requirements for the Draw It or Lose It game application are as follows:

* The game application should be web-based and distributed, which means it can be played on different computing environments and mobile devices.
* The game application should be able to handle multiple users simultaneously.
* The game application should be able to store user data securely.
* The game application should be able to handle user inputs efficiently and respond to them in real-time.
* The game application should provide an enjoyable user experience.

**Design Constraints**

The design constraints for developing the Draw It or Lose It game application in a web-based distributed environment are as follows:

* The game application should be able to handle different computing environments and mobile devices with different hardware and software configurations.
* The game application should be able to handle different network conditions, such as latency, bandwidth, and connectivity.
* The game application should be able to handle different security threats, such as unauthorized access, data breaches, and denial-of-service attacks.

**System Architecture View**

There is nothing required in this section for this project

**Domain Model**

The UML class diagram for the Draw It or Lose It game application shows seven classes: ProgramDriver, SingletonTester, Entity, GameService, Game, Team, and Player. The ProgramDriver class and the SingletonTester class are not related to the game logic, but they are used to start the application and test the Singleton pattern, respectively. The Entity class is a superclass for the Game, Team, and Player classes. The GameService class is responsible for managing the game, which includes creating and updating games, teams, and players, and handling user inputs. The Game class represents a game session, which includes a collection of teams and players. The Team class represents a team, which includes a collection of players. The Player class represents a player, which has a name, a score, and a team. The UML class diagram demonstrates several object-oriented programming principles, such as inheritance, encapsulation, and association. The Entity class is a superclass for the Game, Team, and Player classes, which demonstrates inheritance. The GameService class encapsulates the game logic and manages the game objects, which demonstrates encapsulation. The association between the GameService class and the Game, Team, and Player classes demonstrates association.

**Evaluation**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Operating Platform | Linux | Mac | Windows | iOS | Android |
| Server Side |  |  |  |  |  |
| Does it offer a server-based deployment method? | Yes | Yes | Yes | Yes | Yes |
| Potential licensing costs | Open-source | Paid | Paid | Paid | Paid |
| Client Side |  |  |  |  |  |
| Software development considerations | Ensure compatibility with all web browsers and mobile devices | Ensure compatibility with all web browsers and mobile devices | Ensure compatibility with all web browsers and mobile devices | Ensure compatibility with all web browsers and mobile devices | Ensure compatibility with all web browsers and mobile devices |
| Development Tools |  |  |  |  |  |
| Relevant programming languages | Java, Python, Ruby, PHP | Java, Python, Ruby, PHP | Java, Python, Ruby, PHP | Swift, Objective-C | Java, Kotlin |
| IDEs and other tools | Eclipse, NetBeans, IntelliJ IDEA, Visual Studio Code | Eclipse, Xcode, Atom, Visual Studio Code | Visual Studio, Eclipse, NetBeans, Visual Studio Code | Xcode, AppCode, Visual Studio Code | Android Studio, Visual Studio Code |
| Impact on development team | Familiarity with multiple languages and tools may be required | Familiarity with multiple languages and tools may be required | Familiarity with multiple languages and tools may be required | Familiarity with iOS-specific languages and tools may be required | Familiarity with Android-specific languages and tools may be required |
| Licensing costs | Open-source | Paid | Paid | Paid | Free |

**Server Side:** All of the listed operating platforms can be used for hosting a web-based software application. Linux, Mac, and Windows all offer server-based deployment methods, allowing the website to be hosted on a server. These platforms also offer a variety of open-source and paid options for server operating systems. Linux, being open-source, is the most cost-effective option, whereas Mac and Windows require a paid license.

**Client Side:** To support players on desktop-based operating systems and mobile platforms, the application must be delivered as a modern, responsive HTML interface running inside the web browser. Compatibility with all web browsers and mobile devices is essential. The development process must ensure compatibility with multiple platforms and their respective browser versions. Development teams must be familiar with multiple programming languages and tools to develop applications for different platforms.

**Development Tools:** Java, Python, Ruby, and PHP are commonly used programming languages for web-based software applications, and Eclipse, NetBeans, IntelliJ IDEA, and Visual Studio Code are commonly used IDEs and other tools. For iOS, Swift and Objective-C are the main programming languages, and Xcode, AppCode, and Visual Studio Code are commonly used IDEs and other tools. For Android, Java and Kotlin are the main programming languages, and Android Studio and Visual Studio Code are commonly used IDEs and other tools. Licensing costs vary depending on the specific tool, with some being open-source and others requiring a paid license.

**Conclusion:** In conclusion, all the listed operating platforms can support a web-based software application and can host the Draw It or Lose It game. The development process must ensure compatibility with multiple platforms and their respective browser versions. Development teams must be familiar with multiple programming languages and tools to develop applications for different platforms. Licensing costs vary depending on the specific tool, with some being open-source and others requiring a paid license.

**Recommendations:**

**Operating Platform:** The Gaming Room should use a cloud-based operating platform such as Amazon Web Services (AWS) or Microsoft Azure to host the Draw It or Lose It game. These platforms are flexible, scalable, and can handle high traffic loads. Additionally, they offer features like automatic scaling, load balancing, and high availability, which are essential for an online gaming platform.

**Operating Systems Architectures:** The recommended operating platform should use a virtualized architecture to provide flexibility, security, and manageability. The virtualized architecture allows for the easy deployment and management of applications, as well as the isolation of applications from one another. The platform should also use a containerization technology like Docker to package and deploy the Draw It or Lose It application.

**Storage Management:** To store game data, The Gaming Room should use a distributed NoSQL database system like Apache Cassandra. This database system is scalable, fault-tolerant, and can handle large volumes of data. Additionally, Cassandra offers strong consistency guarantees and can replicate data across multiple data centers, making it an excellent choice for a globally distributed game.

**Memory Management:** The recommended operating platform should use memory management techniques like garbage collection to optimize memory usage and prevent memory leaks. Additionally, the platform should provide tools for monitoring and analyzing memory usage to identify and resolve performance issues.

**Distributed Systems and Networks:** To enable communication between various platforms, The Gaming Room should use a distributed software architecture based on microservices. This architecture allows for the easy deployment and management of small, independent services that can communicate with each other over a network. The services should be designed to handle network connectivity issues and outages and provide graceful degradation when necessary.

**Security:** Security is critical for an online gaming platform, and the recommended operating platform should provide robust security features. The platform should offer features like SSL encryption, access controls, and authentication and authorization mechanisms to protect user information. Additionally, the platform should provide tools for monitoring and analyzing security events to detect and respond to security threats.

**Conclusion:** In summary, The Gaming Room should use a cloud-based operating platform like AWS or Azure, with a virtualized architecture, containerization technology, and a distributed NoSQL database system. The platform should use memory management techniques, microservices architecture for distributed systems, and provide robust security features to protect user information. By following these recommendations, The Gaming Room can create a scalable, secure, and globally distributed Draw It or Lose It game.