

# The Gaming Room

# **CS 230 Project Software Design**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template** 1](#_Toc115077317)

[**Table of Contents 2**](#_Toc115077318)

[**Document Revision History 2**](#_Toc115077319)

[**Executive Summary 3**](#_Toc115077320)

[**Requirements 3**](#_Toc115077321)

[**Design Constraints 3**](#_Toc115077322)

[**System Architecture View 3**](#_Toc115077323)

[**Domain Model 3**](#_Toc115077324)

[**Evaluation 4**](#_Toc115077325)

[**Recommendations 5**](#_Toc115077326)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | <5/20/2023> | Ihab Elrayah | The gaming project is being revised to include entity class implementation, functionality, and a working demo. Various elements such as client support, software development considerations, programming languages and tools, and verifications like security and memory management are being carefully considered  7-1 |

## [Executive Summary](#_sbfa50wo7nsh)

*CTS has been tasked with developing a web-based version of The Gaming Room's game, Draw It or Lose It. The objective is to expand the game's availability to multiple platforms. Key software requirements include support for multiple teams and players, unique game and team names, and the use of unique identifiers for tracking instances.*

*Our proposed solution addresses these requirements by designing a web application that enables team-based gameplay, assigns players to teams, enforces the uniqueness of names, and utilizes unique identifiers. Collaboration with The Gaming Room and regular updates with the technical manager will ensure the application meets expectations.*

*By implementing these features, CTS will deliver a seamless web-based version of Draw It or Lose It, providing an engaging gaming experience across platforms.*

## Requirements

1. *The game application should support multiple teams, allowing for one or more teams to participate.*
2. *Each team should be able to have multiple players assigned to it, facilitating collaborative gameplay.*
3. *Game and team names must be unique to avoid conflicts and allow users to check name availability when selecting a team name.*
4. *Only one instance of the game should exist in memory at any given time to maintain data integrity and prevent conflicts. This can be achieved by implementing unique identifiers for each game, team, and player instance.*

## [Design Constraints](#_2et92p0)

1. *Web-based Distributed Environment: The game application needs to be accessible over the internet and work on different platforms and devices.*
2. *Unique Game and Team Names: Game and team names must be unique to prevent duplication.*
3. *Single Instance of the Game: Only one instance of the game can exist in memory at any time.*

*These design constraints emphasize the web-based nature of the application, the enforcement of unique names, and the management of game instances for a seamless and secure multi-platform gaming experience.*

## [System Architecture View](#_ilbxbyevv6b6)

*The web-based game application for Draw It or Lose It will follow a client-server architecture. The client-side will consist of web browsers, while the server-side will handle the game logic and data storage.*

*Components:*

1. *Client Interface: The user interface for interacting with the game.*
2. *Web Server: Handles client requests and serves resources.*
3. *Game Server: Manages game sessions, teams, players, and game logic.*
4. *Database: Stores game-related information for persistence.*

## [Domain Model](#_8h2ehzxfam4o)

*The UML class diagram represents a game management system. It includes classes such as ProgramDriver, SingletonTester, Entity, GameService, Game, Team, and Player. The classes have attributes, methods, and relationships. The GameService class uses the singleton pattern and manages games. Games consist of teams, and teams consist of players. The diagram demonstrates encapsulation, association, composition, and aggregation, which help organize the classes and fulfill the software requirements efficiently.*

**"The Gaming Room UML diagram. The top of the diagram is labeled as com dot gamingroom. Test boxes are placed in two layers. The first layer has three text boxes and the second layer has four of them. In the first layer, the 'ProgramDriver' textbox points to 'SingletonTester' textbox. The 'ProgramDriver' textbox contains the text 'asterisk main round brackets.' The 'SingletonTester' textbox contains the text 'asterisk testSingleton round brackets.' The arrow between these two text boxes are labeled 'open two angle brackets uses close two angle brackets'. In the second layer, there are 'GameService', 'Game', 'Team', and 'Player' text boxes. The 'GameService' textbox has texts arranged in two layers. The first layer contains games colon List open angle bracket Game close angle bracket, nextGamesId colon long, nextPlayer Id colon long, nextTeamId colon long, and service colon GameService. The second layer contains GameService round brackets, getinstance round brackets colon GameService, addGame open parenthesis name colon String close parenthesis colon Game, getGame open parenthesis id colon long close open parenthesis colon Game, getGame open open parenthesis name colon String close open parenthesis colon Game, getGameCount round brackets colon int, getNextPlayerID round brackets colon long, and getNextTeamId round brackets colon long. The 'GameService' box is connected with the 'Game' textbox with a line labeled 'zero dot dt dot asterisk'.  The 'Game' textbox also contains text in two layers. The first layers contains the text teams colon List open angle bracket Team close angle bracket. The second layer has Game open round bracket id colon long comma name colon String close parenthesis, addTeam open parenthesis name colon String close parenthesis Team, toString round brackets colon String. The 'Game' textbox is connected with the 'Team' textbox with a line labeled 'zero dot dt dot asterisk'. The 'Team' textbox also contains text in two layers. The first layers contains the text players colon List open angle bracket Player close angle bracket. The second layer has Team open parenthesis id colon long comma name colon String close parenthesis, addPlayer open parenthesis name colon String close parenthesis colon Player, and toString round brackets colon String. The 'Team' textbox is connected with the 'Player' textbox with a line labeled 'zero dot dt dot asterisk'. It contains the text Player open parenthesis id colon long comma name colon String close parenthesis and toString round brackets colon String. The 'Game', the 'Team, and the 'Player' boxes point to the 'Entity' textbox in first layer. The 'Entity' textbox contains text in two layers. The first layer has the text id colon long and name colon String. The second layer has Entity round brackets, Entity open parenthesis id colon long comma name colon String close parenthesis, getId round brackets colon long, getName round brackets colon String, toString round brackets colon String.**

## [Evaluation](#_2o15spng8stw)

Using your experience to evaluate the characteristics, advantages, and weaknesses of each operating platform (Linux, Mac, and Windows) as well as mobile devices

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | *-Mac has server-based deployment options, less commonly used then the others.*  *-Integration with Apple technologies can be beneficial.*  *-a commercial operating system, potentially incurring licensing costs.* | *-Linux offers robust server-based deployment methods.*  *-It is open source, reducing costs.*  *-provides stability, security, and flexibility.* | *-Windows provides server-based deployment*  *-Compatibility with Microsoft technologies can be advantageous.*  *-operating systems require licensing, increasing costs.* | *-Mobile platforms require specific considerations for hosting a web-based application.*  *-Development frameworks like React Native or Flutter can cross-platform compatibility.*  *-Mobile platforms have their own app store distribution channels and guidelines.* |
| **Client Side** | *-Ensure compatibility with the latest macOS and Safari.*  *-Evaluate server-based deployment options like Apache.*  *-Consider the potential licensing costs associated with macOS.* | *-Ensure compatibility with popular Linux distributions.*  *- Evaluate server-based deployment options*  *-Linux operating systems are generally open source, reducing licensing costs.* | *-Ensure compatibility with various versions of Windows*  *-Evaluate server-based*  *deployment options like IIS or Apache.*  *-Consider the potential licensing costs associated with Windows.* | *-Develop a responsive HTML interface*  *-Deploy the application on the Apple App Store and Google Play Store.*  *-Consider platform-specific design patterns and performance considerations.* |
| **Development Tools** | *-Programming languages: Objective-C, Swift, C, C++, Python, Ruby, etc.*  *-IDEs and tools: Xcode, JetBrains AppCode, Visual Studio Code, Atom, etc.*  *-Impact on development team: Experience with macOS development tools and languages needed. Multiple teams may be required.*  *-Licensing costs: Some tools are free, but there may costs* | *-Programming languages: C, C++, Python, Java, etc.*  *-IDEs and tools: Eclipse, NetBeans, etc.*  *-Impact on*  *development team: Proficiency in Linux development tools and languages required. Multiple teams may be needed* | *-Programming languages: C#, C++, Java, Python, etc.*  *-IDEs and tools: Visual Studio, JetBrains Rider, Eclipse, etc.*  *-Impact on development team: Proficiency in Windows development tools and languages required. Multiple teams may be needed.*  *-Licensing costs depending on the chosen tools.* | *-Programming languages: Swift, C, Java, etc.*  *-IDEs and tools: Xcode, Android Studio, Visual Studio Code, etc.*  *-Impact on development team: mobile development tools and platform-specific languages needed. Separate teams may be required.*  *-Licensing costs: Xcode and Android Studio are free, costs certain frameworks.* |

## 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**: *To expand Draw It or Lose It to different computing environments, I recommend utilizing a public cloud infrastructure, specifically a Platform-as-a-Service (PaaS) model provided by industry-leading cloud providers such as Amazon Web Services (AWS) or Microsoft Azure. This approach offers scalability, cross-platform compatibility, reliability, availability, cost efficiency, and enhanced security. By adopting the cloud-based operating platform, The Gaming Room can ensure high levels of security for user data and transactions through robust security measures like data encryption and access controls. Additionally, the cloud environment allows for flexible resource allocation, enabling The Gaming Room to scale its resources based on user demand, ensuring a seamless gaming experience across various operating systems and devices.*
2. **Operating Systems Architectures**: *The operating platform architecture recommended for The Gaming Room is based on the Microsoft Azure cloud platform, utilizing the Windows operating system. This choice provides flexibility, scalability, and compatibility across different computing environments. Adopting a microservices architecture, The Gaming Room can enhance functionality and control during the development process. This flexible approach allows for easier maintenance, scalability, and independent deployment of game components. Overall, this architecture ensures a seamless gaming experience, supports expansion to various computing environments, and enables efficient development and management of Draw It or Lose It.*
3. **Storage Management:** *The storage management system recommended for The Gaming Room is Azure Functions, a serverless cloud storage solution. Azure Functions offers cost-effectiveness, ease of setup for different environments, and automatic scalability. It allows The Gaming Room to efficiently manage storage needs, adapt to changing storage requirements, and reduce operational costs. This recommendation aligns with the chosen operating platform and supports the overall software design objectives*.
4. **Memory Management**: *The operating platform, Windows, utilizes effective memory management techniques recommended for the Draw It or Lose It software. It assigns separate virtual address spaces to each process, allowing for up to 4 gigabytes of accessible memory in 32-bit systems and 8 terabytes in 64-bit systems. This ensures efficient memory allocation and protection. Windows also provides heap functions for optimized memory allocation performance and file mapping for effective I/O operations with large data files. By leveraging these techniques, The Gaming Room can ensure efficient memory usage and enhanced performance in their software.*
5. **Distributed Systems and Networks**: *The recommendation for The Gaming Room is to utilize distributed systems and networks, specifically interposes communications (IPC) mechanisms like COM, clipboards, and RPC in Windows, for enabling communication between different platforms in the Draw It or Lose It software. Storing data in Azure, a cloud platform, is also recommended to enhance resilience and protect against network outages. By leveraging these technologies, The Gaming Room can ensure seamless interaction between devices and maintain data availability even during network disruptions. This approach minimizes the impact of connectivity issues and provides a reliable and efficient user experience for Draw It or Lose It across various platforms.*
6. **Security**: *The recommendation for The Gaming Room regarding security is to utilize the security features available in Windows operating platform. Windows offers many robust security measures, including two-factor authentication and coding language support for threat detection. Additionally, the PREfast tool can be used to identify vulnerabilities and bugs in the software. To enhance security during data transmission, utilizing Azure's encrypted IPSec tunnels is advised. Azure also provides malware protection and multi-factor authentication services, further safeguarding user information on and between various platforms. By implementing these security measures, The Gaming Room can ensure the protection of user data and mitigate potential security risks associated with the Draw It or Lose It software.*