<Kevin Alexander>

CS 230 Project Software Design Template

Version 1.0

Table of Contents

Document Revision History

Version	Date	Author	Comments
1.0	<mm/dd/yy>	<Kevin Alexander>	<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 software design problem at hand involves developing a web-based version of the Draw it or Lose it game, where teams compete to guess drawings rendered from a stock image library. The client has outlined specific software requirements that must be met for the game application to function effectively. These requirements include assigning multiple players to each team, accommodating one or more teams per game,ensuring unique game and team names,and restricting the existence of only one instance of the game in memory at any given time. To address these requirements,the proposed solution involves implementing a system that generates unique identifiers for each instance of a game,team, or player. This approach will enable the application to manage multiple teams,prevent naming conflicts,and maintain a single instance of the game in memory. By adhering to these design principles, the software will provide a seamless and efficient gaming experience for users. It is essential for the client to understand that by following this software design approach,we can ensure the smooth operation of the Draw it or Lose it game application,meeting their specified requirements and delivering a high-quality product. This solution sets the foundation for further development and eventual integration with hardware requirements as the project progresses.

Requirements

Business Requirements:

1.Each team will have multiple players assigned to it.

2.A game will have the ability to have one or more teams involved.

3.Game and team names must be unique to allow users to check whether a name is in use when choosing a team name.

Technical Requirements:

1.Only one instance of the game can exist in memory at any given time.

2.Unique identifiers must be created for each instance of a game,team, or player to ensure the uniqueness and proper management of game instances.

Design Constraints

Design Constraints for developing the game application in a web-based distributed environment:

1.Scalability:The application must be designed to handle multiple teams and players concurrently. The system should be able to scale horizontally to accommodate a large number of users without compromising performance.

2. Security:Due to the nature of the game involving multiple users and teams,the application must implement robust security measures to protect user data,prevent unauthorized access, and ensure the integrity of the game sessions.

3.Network Latency: In a distributed environment,network latency can impact the real-time nature of the game. The application should be designed to minimize latency issues to provide a seamless gaming experience for all players.

4.Data Consistency: Ensuring data consistency across distributed systems can be challenging. The application must implement mechanisms such as distributed transactions or eventual consistency to maintain the integrity of game data across all instances.

5.Fault Tolerance: distributed systems are prone to failures, so the application should be designed with fault tolerance in mind. Implementing strategies like redundancy,replication,and graceful degradation can help ensure the availability of the game even in event of failures.

The software requirements provided by the client for the Draw It or Lose it game application have several implications on the application development process.

1.Multiple Players per Team: The requirements for each team to have multiple players assigned to it implies that the application needs to support team management functionalities such as adding,removing,and updating players within a team. This will require implementing user interfaces and backend logic to handle player-team relationships.

2.One or More Teams:The application needs to support multiple teams participating in a game. This

implies that the game logic and user interfaces should be designed to accommodate multiple teams competing against each other simultaneously.

3.Unique Game and Team Names: Ensuring that game and team names are unique adds a constraint on the data management aspect of the application. The system must include validation checks to prevent duplicate names from being used, requiring additional logic for name uniqueness verification during team or game creation.

4.Single Instance in Memory:The requirements for only one instance of the game to exist in memory at any given time suggests the need for proper memory management within the application. This constraint can be addressed by implementing mechanisms to track and implementing mechanisms to track and manage active game instances, ensuring that resources are efficiently utilized and released when no longer needed.

Overall,these design constraints influence various aspects of the application development process ,including data management,user interface design,memory handling,and overall system architecture. Adhering to these requirements will help ensure that the Draw It or Lose It game application meets the client's expectations.

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

Classes relate to each other:1.ProgramDriver:-Contains the main() method. 2.Game Service:-games:List of Game objects. -nextGameId:long value for the next gameID.-nextTeamId:long value for the next team ID.-Methods:-addGame(name:String):Adds a new Game object to the

service.-getGame(id:long):Retrieves a Game object by ID.-getGame(name:String):Retrieves a Game object by name.-getNextGame():Returns the next game ID.

3.SingletonTester:-Methods:-testSingleton():Tests the Singleton pattern.

4.Game:Attributes:-teams:List of Team objects.-Methods:-Game(id:long, name:String):Constructor to create a new Game object.-addTeam(name:String):Adds a new Team to the Game.-getNextTeam():Returns the next team ID. 5.Team:Attributes:players:List of Player objects.-id:long value for the team ID.

The UML diagram demonstrates several object-oriented programming principles:

1.Encapsulation:Classes like Game,Team,Entity,and Player encapsulation their data(attributes)and behavior(methods)within themselves.

2.Inheritance:There is an inheritance relationship between ‘Entity’ and Team where ‘Team’ where ‘Team’ inherits from ‘Entity’. This relationship,indicating that a ‘Team’ is a type of’ Entity’.

3.Association:The diagram shows associations between classes like ‘Game’ and ‘Team’,’Team' and ‘Player’,and ‘Game’and ‘Entity’. These associations represent relationships between the classes,such as a ‘Game’ having multiple ‘Team’ instances.

4.Aggregation/Composition:The diagram shows aggregation/composition relationships between ‘Game’ and ‘Team’ and ‘Team’ and ‘Player’. This relationship implies a has a relationship,indicating that a “Game’ has multiple ‘Team’ instances,and a ‘Team’ has multiple ‘Player’ instances.

5.Polymorphism:While not explicitly shown in the diagram,polymorphism can be inferred from the methods like ‘toString()’ being overridden in subclasses like ‘Team’ and ‘Player’. This allows different classes to provide their own implementation.

To fulfill the software requirements efficiently using object-oriented principles,we can identify the following key points from the diagram:

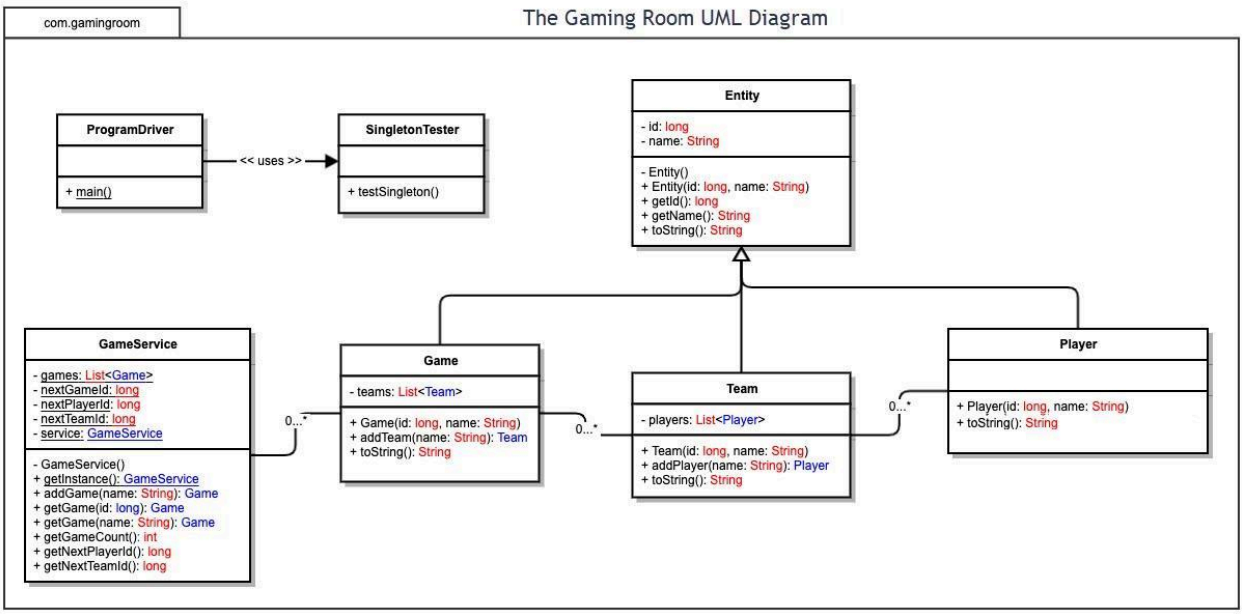
1.Encapsulation:Classes like Game, Team,Player,and Entity encapsulate their data and behavior within themselves. This helps in maintaining data integrity and controlling access to the data.

2.Inheritance:The diagram shows inheritance relationships such as Game extends Entity,Team extends Entity. This allows for code reuse and promotes a hierarchical structure among classes.

3.Polymorphism:Polymorphism allows objects of different classes to be treated as objects of a common superclass.

4.Single Instance in Memory:The requirements for only one instance of the game to exist in memory at any given time suggests the need for proper memory management within the application. This constraint can be addressed by implementing mechanisms to track and manage active game instances,ensuring that resources are efficiently utilized and released when no longer needed.

Overall,these design constraints influence various aspects of the application development process,including data management,user interface design,memory handling,and overall system architecture. Adhering to these requirements will help ensure that the Draw It or Lose It game application meets the client’s expectations.



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: Stability:macOS Server is known for its stability and reliability,making it suitable for hosting critical web-based applications. Integration with Apple Ecosystem:macOS Server seamlessly integrates with other Apple products and services,which can be beneficial if the clients environment is predominantly Apple-based. Advantages:User-Friendly Interface:macOS Server provides a user-friendly interface that simplifies the management of server settings and configurations. Weakness:Limited Hardware Compatibility:macOS Server is designed to run on Apple hardware,which may limit choices.	Characteristics: Open Source:Linux is open-source software,which means that the source code is freely available for users to modify and distribute. Stability:Linux is known for its stability and reliability,making it a suitable choice for hosting critical applications. Advantages:Cost-effective:Linux is free to use and does not require expensive licensing fees,making it a cost-effective solution for hosting web-applications. Weaknesses: Learning curve:Linux may have a steeper learning curve for users who are unfamiliar with the operating system,which could require additional training or expertise..	Characteristics: Windows Server provides a stable and reliable platform for hosting web applications.It offers robust security features to protect the application and data. Advantages:Integration with other Microsoft products and services,such as Azure cloud services,for seamless development. Active Directory integration for user authentication and access control. Weakness:Licensing costs associated with Windows Server can be higher compared to open-source alternatives.	Characteristics: Limited Resources:Mobile devices have limited processing power,memory,and storage compared to traditional servers,which can impact the performance of the web application. Connectivity:Mobile devices rely on wireless networks,which may not always provide stable and high-speed connections,affecting the responsiveness of the applications. Advantages: Proximity:Mobile devices can provide low-latency access to the application for users,especially in scenarios where they are close to the server,enhancing the overall user experience. Weaknesses: Reliability:Mobile devices may not offer the same level of reliability as traditional servers,as they are prone to battery drain,network disruptions,and other mobile-specific issues.

Client Side	Cross-Platform Compatibility: Ensure that the game application is developed using technologies that support deployment on Mac operating systems. Consider using frameworks like Electron,React Native,or Unity that allow for building applications that can run on multiple platforms including Mac. 2.User Interface Design:Design the user interface of the game application to be responsive and compatible with Mac devices. Consider Mac-specific design guidelines to provide a seamless user experience for Mac users.	1.Cost:Developing for Linux can be cost-effective as it is an open-source platform. However,the cost may vary based on the need for specialized expertise, and any licensing fees for third-party tools or libraries. 2.Time: Developing for Linux may require additional time for testing and debugging across different distributions and versions, Ensuring compatibility with various Linux distributions and architectures can also impact the development timeline. 3.Expertise: Developers with expertise in Linux development, including knowledge of different distributions, package management systems,and	Cost:Developing for multiple types of clients,include Windows,may increase development costs due to the need for additional resources,tools, and testing across different platforms. However, leveraging cross-platform development frameworks or tools can help reduce costs by allowing code sharing and minimizing platform-specific development efforts. 2.Time:Supporting multiple types of clients,like Windows,may extend the development timeline as each platform may have unique requirements, design guidelines,and testing needs. It is essential to allocate sufficient time for platform-specific development,	Cost:Development for multiple types of clients,especially mobile devices,can increase development costs due to the need for testing,optimization, considerations for various screen sizes and potentially different design considerations for various screen sizes and operating systems. The cost will depend on the complexity of the game,the number of platforms supported, and the level of customization required for each platform. 2.Time:Supporting multiple types clients can increase the development time as each platform may require specific optimizations, adjustments,and testing to ensure a consistent user experience across devices. Time estimates should include development,testing ,debugging,and deployment phases for each platform. Expertise: 3.Expertise:

	3.Testing and Quality Assurance:Allocate time and resources for thorough testing on Mac devices to ensure that the game functions correctly and optimally on Mac operating systems. Consider setting up a testing environment with Mac machines to perform compatibility testing. 4.MacOS App Store Compliance:If the game application is intended for distribution through the Mac App Store, ensure that it meets the necessary guidelines and requirements set by Apple for submission and approval.	system libraries will be essential. Understanding Linux-specific development tools and best practices is crucial for efficient development. 4.Portability: Consider using cross-platform development framework or tools that support Linux to enhance portability across different client types. This can help streamline development efforts and reduce the need for platform-specific code. 5.Security:Linux systems are known for their security features,so ensuring that the game application follows best security practices on Linux is important. This includes implementing secure communication protocols,data encryption,and access control mechanisms. 6.Testing:	testing,and optimization to ensure a consistent user experience across different client types. 3.Expertise: Developing for Windows requires expertise in Windows development technologies, framework, and best practices. It is important to have developers with the necessary skills and experience in Windows application development to ensure the successful implementation of the game application on the Windows platform.	Developing for multiple types of clients requires expertise in different programming languages, framework,and tools specific to each platform.Developers need to be proficient in mobile app development for various operating systems(iOS,Android ,etc.)and have experience in optimizing performance and user interface for different screen sizes and resolutions. 4.Cross-Platform Development Tools:Consider using cross-platform development tools and framework such as React Native,Flutter, Xamarin,or Unity to streamline development for multiple platforms. These tools can help reduce development time and costs by allowing code sharing across different platforms.

	5. Development Expertise: Ensure that the development team has expertise in Mac application development or access to resources with experience in developing applications for Mac devices. Consider hiring developers with experience in MacOS development if needed. 6.Cost Considerations: Developing and testing for Mac compatibility may incur additional costs in terms of resources,time,and expertise, Factor in these costs during the project planning phase to ensure that the budget accommodates the development for Mac.	Comprehensive testing on different Linux distributions,desk top environments,and hardware configurations is necessary to ensure the game application functions correctly across various clients types. Automated testing tools can help streamline this process.		5.User Experience(UX)Design:Designing the user interface and experience to be responsive and adaptive to different screen.

Development Tools	1.Programming Languages: Frontend;HTML, CSS,,JavaScript(for rendering images, user interface, and game logic). 2.IDE(Integrated Development Environment): Visual Studio Code:A popular and versatile IDE for web development that supports HTML,CSS JavaScript,and Node.js. 3.Express.js:A web application framework for Node.js that can be used to build the backend of the game application. Socket.IO:For real-time communication between the server and clients,enabling features like live updates and multiplayers interactions. MongoDB:A NoSQL database that can be used to store game,team,and player data. Postman:For testing API endpoints and ensuring proper communication between frontend and backend.	1.Programming Languages:Frontend:HTML,CSS, JavaScript(with frameworks like React,Angular,or Vue.js for interactive UI). Backend:Node .js with Express.js for server-side logic. Database: MongoDB or MySQL for storing game,team,and player information. 2.IDE and Tools:IDE: Visual Studio Code,Sublime Text, or Atom for coding. Version Control:Git for managing source code. Package Manager:npm(Node Package Manager)for managing dependencies. API Development:Postman for testing APIs. Deployment Tools: Docker for containerization, Nginx for web server, and PM2 for process management.	Programming Languages: Backend:You can use languages like Python with frameworks like Django or Flask for the server-side logic. Frontend:For the client-side interface,you can use HTML,CSS,and JavaScript with frameworks like React or Angular. IDE(Integrated Development Environment). PyCharm:A popular IDE for Python development that provides a range of tools for efficient coding. Visual Studio Code:A versatile code editor that supports various languages and extensions for web development. 3.Database:SQLite :A lightweight database that can be used for storing game,team,and player information.	Programming Languages:Frontend :HTML 5,CSS,JavaScript(for rendering images,user interface,and game logic). Backend:Node.js(for server-side logic,handling game data,and communication with the frontend). 2.Tools and IDEs: Frontend Development:IDE: Visual Studio Code,Sublime Text,or WebStorm. Frameworks/ Libraries:React.js or Vue.js for building interactive user interfaces. Graphics Library:Three.js or PixJS for rendering images and animations. Version Control:Git for managing source code. Backend Development: IDE:Visual Studio Code,Atom,or JetBrains WebStorm. Framework:Express.js for building RESTful APIs and handling server-side logic. Database:MongoDB or MySQL for storing game data and user information.

	Git and GitHub:Version control system and repository for managing code changes and collaboration. Heroku:A cloud platform that can be used for deploying and hosting the web application.		4.Version Control:Git:Utilize Git for version control to manage the codebase efficiently. 5.Other Tools:Django Rest Framework:If using Django,this can help in building ResTful APIs for communication between the frontend and backend. React or Angular CLI:For setting up the frontend application structure and managing dependencies. Postman:To test APIs and ensure proper communication between frontend and backend components.	Authentication: JSON Web Tokens(JWT)for use authentication and authorization. Testing:Mocha,Chai, or Jest for backend testing. Mobile Deployment: Cross-Platform Development: Consider using frameworks like React Native or Flutter for building the mobile app for both iOS and Andriod platforms.

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 facilitate the development of the web-based version of the Draw It or Lose It game application and to allow for expansion to other computing environments, a suitable operating platform recommendation would be to use a cloud-based platform recommendation would be to use a cloud-based platform such as Amazon Web Services(AWS) or Microsoft Azure. Cloud platforms offer scalability,flexibility and ease of deployment across different environments. They provide the necessary infrastructure and services to support web applications,ensuring high availability and performance.Additionally,cloud platforms offer tools and services for managing instances,databases,and other resources efficiently.

   By leveraging a cloud-based platform,The Gaming Room can easily deploy and manage the Draw It or Lose It game application across various computing environments,including web browsers, mobile devices,and other platforms,while ensuring optimal performance and scalability.

2. **Operating Systems Architectures**: Based on the software requirements provided for the Draw It or Lose It game application,an appropriate operating platform that would allow The Gaming Room to expand the game to other computing environments would be a cloud-based platform. Cloud platforms provide scalability,flexibility and accessibility across different devices and operating systems. The chosen operating platform architecture for the Draw It or Lose It game application could be based on a microservices architecture deployed on a cloud platform such as Amazon Web Services(AWS) or Microsoft Azure. This architecture would allow for the following components:

   Game Service:Responsible for managing game instances,including creating unique identifiers for each game, team and player.It ensures that only one instance of the game exists in memory at any given time. Team Service:Managers teams within the game,assigns players to teams,and

ensures that team and game names are unique. Player Service:Handles player-related functionalities,such as player profiles,scores,and interactions within the game.Cloud Infrastructure:Utilizes cloud services for hosting the game application,managing resources,and ensuring scalability and availability across different computing environments.

By adopting a microservices architecture on a cloud platform,The Game Room can easily expand the Draw It or Lose It game application to various computing environments while maintaining the required unique identifiers and ensuring a seamless gaming experience for users.

3. **Storage Management**: Database as a Service(DBaas):To manage the storage requirements of the game application,a cloud-based database service like Amazon RDS(Relational Database Service)or Azure SQL Database can be used. These services provide managed database instances that can scale based on demand,offer high availability,and ensure data security and integrity. By leveraging a cloud-based platform with a microservices architecture and utilizing a reliable storage management system like a DBaas,The Gaming Room can streamline the development and deployment of the Draw It or Lose It game application while ensuring scalability,performance and data management.
4. **Memory Management**: Techniques:The recommended operating platform will utilize memory management techniques such as caching to optimize performance. Caching frequently accessed data in memory can reduce the need for repeated database queries,improving response times and overall user experience. Additionally,the platform should efficiently manage memory allocation and deallocation to prevent memory leaks and ensure optimal resource utilization.Utilizing a cloud-based operating platform with appropriate architecture,storage management system,and memory management techniques.The Gaming Room can streamline the development of the Draw It or Lose It game application and prepare.
5. **Distributed Systems and Networks**: To enable communication between various platforms for the

Draw It or Lose It game,a service-oriented architecture(SOA)approach can be implemented. This involves breaking down the application into smaller,independent services that communicate over a network. Each service can perform specific functions and interact with other services through well-defined interfaces. Network Connectivity:To ensure seamless communication between devices and platforms,a robust network infrastructure is essential. The Draw it or Lose It game application can utilize RESTful APIs (Representational State Transfer) for communication between different components. This allows for easy integration and interoperability across various platforms.Dependencies and Outages:It is crucial to consider dependencies between components within the distributed systems and networks to ensure reliability and fault tolerance. Implementing redundancy,load balancing,and failover mechanisms can help mitigate outages and ensure continuous availability of the game application.

6. **Security**: To protect user information on and between various platforms,you can implement security measures such as encryption for data transmission(using HTTPS protocol),secure authentication mechanisms(e.g,OAuth,JWT),and data validation to prevent common security vulnerabilities like SQL Injection and cross-site scripting(XSS). Additionally,you can implement role-based access control to ensure that only authorized users have access to sensitive information. Considering the user protection and security capabilities of the recommended operating platform(web-based),It is essential to stay updated with security patches regularly,conduct security audits,and implement secure coding practices to mitigate potential security risks. By following these measures,you can ensure that user information is protected and secure different platforms while playing Draw It or Lose It.

Evaluation

To evaluate the characteristics, advantages,and weaknesses of deploying the Draw It or Lose It game application on the three traditional operating platforms(linux),

Characteristics: Linux is an open source operating system known for its stability,security,and flexibility. It offers a wide range of distributions tailored to different needs,such as Ubuntu, CentOS,and Debian. Linux supports a variety of programming languages and tools commonly used in software development. Advantages linux is free to use and distribute,which can result in cost savings for businesses. Customizability allows for extensive customization to meet specific requirements of the game application. Security linux is known for its robust security features,making it a reliable choice for deploying applications.Stability linux systems are known for their stability and uptime ,which is crucial for a gaming application. Compatibility Some software may not be compatible with linux,which could require additional effort for integration. User Interface linux desktop environments may not be user-friendly as other platforms,which could impact.For Linux operating platform offers a server-based deployment method where the website can be hosted. Linux is well-known for its robust server capabilities and is widely used for hosting websites and web applications. It provides various server software options such as Apache,Nginx,and others that can handle high traffic loads efficiently.Linux also offers strong security features and is highly customizable,making it a popular choice for server-side deployments. Overall,Linux is a reliable and scalable platform for hosting web-based applications like Draw It or Lose It.

For a web-based application running on a server-side configuration on Linux, the potential licensing costs for the server operating system would be minimal or zero. Linux is an open-source operating system,which means it is freely available to download,use,and distribute. There are no licensing fees associated with using Linux as the server operating system,making it a cost-effective choice for hosting websites and scaling up to thousands of players. This can be a significant advantage for clients like The Gaming Room as it helps reduce overall operational costs and allows for more flexibility in deploying and managing the application.

Expanding the Draw It or Lose It gaming app to multiple platforms and supporting players on Linux,the client can consider developing a modern responsive HTML interface that runs inside web browsers on desktops. This approach will allow players on linux systems to access the game through their web browser,providing a platform-independent solution. By utilizing web technologies such as HTML,CSS,and JavaScript,the client can create responsive and different screen sizes and devices. This will enhance the accessibility and usability of the game across various desktop platforms,offering a seamless gaming experience for players.

Mac Operating platform,when considering server-side deployment for a web-based application,Mac OS does offer server-based deployment methods. Mac OS provides tools and technologies such as macOS Server,Apache web server, and other third-party server software that can be used to host websites and

scale up to accommodate thousands of players. Mac OS is known for its stability and security,which are crucial factors for hosting web applications. However,Mac OS may have limitations in terms of scalability compared to other server platforms like Linux or Windows Server. It is important to consider the specific requirements of the game application and the expected player base when choosing Mac OS for server-side deployment.

Mac operating systems used as a server-side configuration for hosting a web-based application like Draw it or Lose It,the potential licensing costs to the client, The Gaming Room,would typically involve purchasing a license for macOS Server. As of my last update,macOS Server is available for a one-time fee of around $19.99 on the Mac App Store. This license allows for hosting services such as websites,mail,calendars,contacts,chat,Time Machine,VPN,and more on a Mac computer running nacOS. Additionally,there may be additional costs associated with hardware requirements.maintenance,and support services depending on the scale and complexity of the application and server setup. It's important for the client to consider these factors when evaluating the overall cost of deploying the game application on a Mac server platform.

Expanding the Draw It or Lose It gaming app to multiple platforms,the client can consider implementing a web-based solution using a modern responsive HTML interface. This approach will allow players on desktop platforms like Mac to access the game through a web browser,providing a seamless and consistent user experience across different devices. By leveraging web technologies,such as HTML.CSS and JavaScript,the client can ensure compatibility with a wide range of devices and operating systems without the need for platform-specific development. This approach aligns with the goal of supporting players on desktop platforms while maintaining a distributed environment for the gaming application.

For the Windows operating platform,when considering server-side deployment for a web-based application like Draw It or Lose It,Windows Server operating systems such as Windows Server 2019 would be the suitable choice. Windows Server provides a robust platform for hosting websites and scaling up to accommodate thousands of players. It offers features like Internet Information Services(IIS) for web hosting,support for various web technologies and scalability options to handle increased traffic and user load.Characteristics of Windows is a widely used operating system known for its compatibility with a wide range of software and hardware. It is commonly used in enterprise environments and for hosting various types of applications. Advantages of Windows provides good support for hosting web applications through Internet Information Services(IIS).IT offers integration with other Microsoft Technologies and tools,making it suitable for organizations already invested in the Microsoft ecosystem.Weakness Of Windows servers may require more frequent updates and maintenance compared to Linux servers. They are also perceived to be more vulnerable to security threats,although this can be mitigated through proper configuration and security measures.

Windows Server Server operating system,the potential licensing costs for the client. The Gaming Room.can vary depending on the specific edition and licensing model chosen. Windows Server Standard edition is suitable for physical or minimally virtualized environments. The licensing cost is typically based on the server. As of the time of writing,the approximate cost for a 16-core license is around $972 USD. Windows Server Datacenter edition is designed for highly virtualized environments . As of the time of writing,the approximate cost for a 16-core license is around $6,155 USD. Client Access Licenses(CALs) in addition to server license,client access License may be required for each user or device accessing the server. The cost of CALs can vary based on the number of users or devices that need access. Subscription-based Licensing Microsoft also offers subscription-based licensing options such as Windows

Server Subscription or Azure Hybrid Benefit for Windows Server,which may provide cost savings based on usage and requirements.

Expanding the gaming app to multiple platforms and supporting players on Windows,the client can consider implementing a modern responsive HTML interface that runs inside a web browser on desktops. This approach will enable users on Windows devices to access the game seamlessly without the need for a dedicated Android application. By utilizing responsive design techniques the utilizing responsive design techniques the interface can adapt to different screen sizes and resolutions,providing an optimal user experience across various desktop devices. Additionally.leveraging web technologies for the client-side implementation can facilitate easier maintenance,updates,and cross-platform compatibility.

The iOS Platform the characteristics of the iOS is the operating system developed by Apple for its mobile devices. It has a closed ecosystem with strict guidelines for app development and deployment. Advantages Apple provides robust server-side deployment options through services like iCloud and Apple Developer Program. Offers secure and reliable hosting solutions for web applications. Weaknesses limited flexibility compared to other platforms in terms of server-side configurations. Higher cost associated with Apple's hosting services.

The iOS(Apple) iOS is known for its closed ecosystem,security,and user experience. Advantage in iOS provides a stable and secure environment for hosting web applications. Weaknesses licensing costs for server-side operating systems on iOS can be relatively high compared to other platforms. Potential Licensing Costs client may need to consider the costs associated with macOS Server for hosting web applications on iOS devices.

Android Platform is an open-source operating system developed by Google for mobile devices. It allows for more customization and flexibility in app development. Advantages Google Cloud Platform provides scalable and cost-effective server-side deployment options. Offers a wide range of hosting solutions suitable for web applications. Weaknesses the security concerns due to the open nature of the platform. Fragmentation across different devices and versions may require additional testing and optimizations.

Android is an open-source platform with a wide range of devices and customization options. Advantages android offers flexibility and scalability for hosting web applications. Weaknesses android may have more variability in terms of server-side configurations and support. Potential Licensing Costs clients could explore using Linux-based server operating systems,which are often open-source and have lower licensing costs compared to proprietary systems.

The client's gaming app expands to multiple platforms using various software patterns in a distributed environment. The client-Side Development supports players on multiple platforms,the application should be developed as a modern,responsive HTML interface that runs inside web browsers. This will ensure compatibility with desktops and various mobile devices. Responsive Design implements a responsive design approach to ensure that the gaming app adapts to different screen sizes and orientations,providing an optimal viewing experience across devices. Cross-platform Compatibility utilizes technologies like HTML 5,CSS3,and JavaScript to create a cross-platform gaming experience that can be accessed on desktop,smartphones,and tablets without the need for platform-specific development.User Experience(UX) focus on creating an intuitive and engaging user experience that is

consistent across all platforms. Consider factors such as touch interactions for mobile devices and mouse/keyboard inputs for desktop users.

The technical requirements for expanding the gaming app to multiple platforms in a distributed environment will have several impacts on the development team. These are key points. The Increased Complexity:Developing for multiple platforms and implementing distributed systems can significantly increase the complexity of the project. The team will need to consider compatibility issues,network communication and synchronization between different components. Specialized skill sets the development team may need to acquire or enhance their skills in areas such as cross-platform development,distributed systems,and network programming. This may require additional training or hiring of specialized developers. Collaboration challenges working in a distributed environment may introduce challenges related to collaboration and communication among team members. Clear communication channels and project management tools will be essential to ensure smooth coordination. Testing and Debugging testing the application across multiple platforms and in a distributed environment will require thorough testing strategies. The team will need to invest time in testing and debugging to ensure the app functions correctly on all target platforms. Performance Optimization of the app for performance in distributed environments will be crucial. the team will need to focus on efficient data transfer.load balancing,and scalability to ensure a smooth user experience. Security Considerations implementing security measures to protect data in a distributed environment is paramount. The team will need to prioritize security practices such as encryption authentication and access control to safeguard the app and user data.

There are no costs associated with Linux as it is an open-source operating system. Mac and Windows,on the other hand, may have licensing costs associated with their development tools. It is important to consider these costs when deciding on the platform for deploying the game application software.