

<Draw it or Lose it>

**CS 230 Project Software Design Template**

Version 2.0

**Table of Contents**

**Document Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| 1.0  2.0  3.0 | <03/14/23>  <04/01/23>  <4/16/23> | <Matthew Leclerc> | <Initial Creation.> <Project Two Submission Client Side/Server Side/Development Tools>  <Project Three Submission> |

**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 challenge is to develop a web-based edition of Draw It or Lose It game, which allows multiple teams and players to compete in guessing a puzzle created from a collection of pre-drawn images. To accomplish this, we suggest creating a web application that stores and manages game, team, and player data in a backend database. The front-end interface will interact with the backend through RESTful APIs, ensuring effortless scalability and maintenance. Each game and team will have unique names to differentiate them from others.

**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**

There are several design limitations to consider when creating a web-based distributed environment as part of the application development process. These constraints include:

**Security:** As the game application will exist in a web-based environment, it is vulnerable to security threats like injection attacks, cross-site scripting, and cross-site request forgery. To mitigate this risk, we will follow secure coding practices, ensure that input is validated, and utilize secure communication protocols such as HTTPS.

**Network Latency:** Network latency has the potential to impact the performance of the game application, resulting in slow or unresponsive behavior and, ultimately, a negative user experience. To address this issue, we will create the application in such a way as to limit the number of network requests and optimize data transfer, thereby mitigating this constraint.

**Scalability:** As the popularity of the game may grow over time, it may be necessary to scale the application either horizontally or vertically. To address this constraint, we will employ cloud hosting providers that offer scalable infrastructure, as well as implement load balancing techniques to ensure high availability and optimal performance.

**Data Persistence:** To ensure that the game's data, including game, team, and player information, is preserved even when the application is not running, we will utilize a backend database to store and manage the data.

**Browser Capability:** The game application needs to be compatible with various web browsers and devices. To achieve this, we will test the application across multiple browsers and devices and use responsive design techniques to guarantee compatibility.

**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 Entity class establishes a relationship between the Game, Team, and Player classes, meaning that they inherit or receive information from the Entity class. This is represented through inheritance in the UML diagram, with each class sharing common attributes such as "name" and "id" by making Entity the superclass. In terms of their relationships, Team and Player have a "has a" type, while Game has a Team and GameService has Games. In UML, we refer to this as aggregation (HAS-A), where an instance of one class has a reference to an instance of another class. The diagram shows that GameService has a reference to Games, Games has a reference to Team, and Team has a reference to Player.



**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 servers are built on Unix, making them reliable and stable.  **Advantages:**  Mac servers are compatible with a wide range of programming languages and frameworks, including Ruby on Rails, Node.js, and PHP.  **Disadvantages:**  Mac servers have limited scalability compared to cloud-based hosting options, which can grow and shrink on demand.> | < Linux is an open-source operating system, which means that it is free to use and modify. This makes it a cost-effective option for hosting web-based software applications.  **Advantages:** Linux is highly efficient, which means that it can run on older hardware, reducing the need for expensive server upgrades.  **Disadvantages:**  Linux may not be compatible with all software applications, which can limit its usefulness for some users.  > | < Windows Server supports Microsoft SQL Server as a database management system, as well as other popular database systems like MySQL and PostgreSQL.  **Advantages:** It offers excellent integration with other Microsoft products, including Office, SharePoint, and Exchange.  **Disadvantages:** It may require more powerful hardware to run smoothly compared to Linux or other operating systems.> | <Mobile devices are characterized by their limited processing power and screen size, which can affect the performance and user experience of web-based software applications. Additionally, these devices often rely on mobile networks or Wi-Fi connections to access the internet, which may be slower and less dependable than wired connections.  **Advantages:** Mobile devices offer a highly portable and convenient platform for users to access web-based software applications on-the-go.  **Disadvantages:** Mobile devices have smaller screens, which can make it challenging to display complex web-based software applications that require a lot of user input or interaction.> |
| **Client Side** | < Firstly, developing software for multiple types of clients requires a higher level of expertise and experience, which may result in higher development costs. Developers need to have knowledge of multiple programming languages and platforms to create software that is compatible with Mac, iOS, and other platforms. This can lead to increased development time and cost.> | < The expense of creating software for Linux may fluctuate based on the intricacy of the project and the proficiency of the development team. Nevertheless, utilizing open-source tools and platforms can aid in decreasing the total cost of development. Designing software that can accommodate various client types on Linux may necessitate more time than developing software for a single client. Creating software for Linux demands a high degree of proficiency in Linux-based technologies and tools, such as programming languages like C and Python, and development frameworks like GTK and Qt. > | < The cost of developing software for Windows can vary depending on factors such as project complexity, team size, and required expertise. Developing software that can support multiple types of clients on Windows can be more expensive due to the need to create separate applications for each client type. Additionally, the development process may take longer due to the added complexity of supporting multiple clients. Windows requires a high level of expertise in Windows-based technologies and tools, such as programming languages like C# and .NET, and development frameworks like Windows Presentation Foundation (WPF) and Universal Windows Platform (UWP). > | < Developing software for mobile devices incurs costs that vary based on the complexity of the project, team size, and the level of expertise required for mobile application development. Supporting multiple types of clients on mobile devices can extend the development process compared to developing for a single platform. The development team must ensure the application is compatible with multiple operating systems and device types, increasing development time. Developers must have expertise in mobile application development for various platforms, such as Android and iOS, and programming languages like Java, Kotlin, Swift, and Objective-C.> |
| **Development Tools** | <Swift is a modern programming language created by Apple for building software applications for their operating systems such as iOS, macOS, and others. This language is fast, safe, and easy to read and write.  Objective-C is another programming language developed by Apple and commonly used for building software applications for macOS and iOS. For software development on Mac, developers commonly use  Xcode, an integrated development environment (IDE) developed by Apple. Xcode is a powerful tool that includes features for writing, testing, and debugging software applications for macOS, iOS, watchOS, and tvOS.> | <**C:** A popular and widely-used low-level programming language used to build Linux system-level software and applications.  **C++:**  A powerful, object-oriented programming language used to build high-performance applications and system-level software on Linux.  **Python:**  A high-level programming language known for its ease of use, readability, and versatility. It is often used to develop scripting and automation tools on Linux.  **Java:**  A popular, platform-independent programming language that can be used to develop applications on Linux. | <**C#:** C# is a modern object-oriented programming language developed by Microsoft. It is commonly used for developing Windows desktop applications, web applications, and games.  **.NET Framework:** .NET Framework is a software framework developed by Microsoft for building Windows applications. It includes a large library of pre-built components and tools for developing Windows applications.  **Visual Studio:** Visual Studio is an integrated development environment (IDE) developed by Microsoft for developing Windows applications. It includes a suite of tools for writing, testing, and debugging software.> | <**Flutter:** Flutter is an open-source mobile application development framework developed by Google. It allows developers to build mobile applications for both iOS and Android using the Dart programming language.  **Java and Kotlin**: For developing Android applications, Java has been the primary programming language. However, Kotlin is also gaining popularity as an alternative language for developing Android applications.  **Android Studio:** Android Studio is the primary IDE used for developing Android applications. It includes a suite of tools for writing, testing, and debugging software. |

**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**: < I recommend starting with Windows devices as the primary platform for the game. Windows has a larger market share and a wider range of software available, making it easier to develop and expand the game. Additionally, there are many IDEs available for Windows, reducing the need for specialized expertise and potentially lowering costs.>
* **Operating Systems Architectures**: < The Windows kernel provides a layer of abstraction between the hardware and software, allowing programs to run on different hardware platforms with minimal changes. Windows also includes a graphical user interface (GUI) called the Windows Shell, which allows users to interact with the system using a mouse, keyboard, or touch screen.>
* **Storage Management**: < Windows Storage Spaces is a built-in feature in Windows that allows users to group physical disks into storage pools, which can then be used to create virtual drives called storage spaces. This allows for flexible storage allocation and management, as well as data redundancy through mirroring or parity.>
* **Memory Management**: < Windows, uses virtual memory management techniques to manage memory for applications. Virtual memory is a technique where the operating system uses a portion of the hard drive as a temporary storage area to store data that is not currently being used in physical memory (RAM). Windows also provides a feature called the Memory Manager, which is responsible for managing memory allocation for applications. The Memory Manager allocates memory to applications on a demand basis and releases the memory when it is no longer needed.>
* **Distributed Systems and Networks**: <To achieve cross-platform communication, a distributed software system can be implemented along with a network connecting the devices. One option is to use a client-server architecture, in which the server manages the game and communicates with clients on various platforms. The clients can access the server through the internet or a local area network (LAN). Dependencies between the components within the distributed systems and networks, such as connectivity and outages, must be considered during the design and development process.>
* **Security**: < To ensure security for user information, encryption should be implemented for data at rest and in transit. User authentication, strong password policies, access control, and firewall should also be utilized. Additionally, the recommended operating system platform should have built-in security features such as antivirus software, firewall, and secure boot.>