

<Name-of-Software-Application>

**CS 230 Project Software Design Template**

Version 1.0

**Table of Contents**

**Document Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| 1.0 | 08/13/2023 | Eduardo Cruz | Update of recomendations |

**Executive Summary**

The Gaming Room wants to develop a web-based game that can run on multiple platforms. The game will be called “Draw It or Lose It” and is currently only available on android. The purpose of this game is multiple teams consisting of several people going four rounds at a minute each. When a picture is pulled from a library of images one team guesses till time runs out. If not answered each opposing team member gets to answer till 15 seconds runs out.

**Requirements**

*The requirements needed to follow while writing the code and software. While this is only the game aspect, we still need to look at application development. The Gaming Room would like this to run on all devices. This means we already have it on android but need to work it into another mobile device. Along with machines like Windows, Linux, and Apple. To do this we will need to find a way to either re-write the code in swift for (Apple devices) or come up with a way to use existing code to be run on other devices by inheriting other languages. Like when we use multiple computer languages together to make stronger code.*

**Design Constraints**

* Needs one or more teams involved
* Each team has multiple people
* Game and Team names must be unique to allow users to check whether the name is in use or free
* Only one instance of the game can exist at any time.
* Must run on multiple platforms

**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 represents the domain model of the game application. It consists of several classes, including GameService, Game, Team, and Player, interconnected to fulfill the software requirements.

The Entity class serves as the base class for all entities in the application, providing common attributes like ID and name. This demonstrates the principle of inheritance, allowing other classes to inherit these attributes and behaviors. By utilizing the Entity class, we avoid code duplication and ensure consistency across entities.

The GameService class acts as the central service for managing games. It maintains a list of games and provides methods to add games, retrieve games by ID or name, and retrieve the total game count. This class follows the Singleton design pattern, ensuring a single instance is accessible throughout the application. The SingletonTester class <<uses>> the ProgramDriver class to test the singleton behavior.

The Game class represents a game in the application. It maintains a list of teams participating in the game and provides methods to add teams and retrieve a string representation of the game. The Team class represents a team within a game and maintains a list of players. It provides methods to add players and retrieve a string representation of the team. The Player class represents an individual player and provides a method to retrieve a string representation.

These classes demonstrate the principle of composition, where complex objects are built by combining simpler objects. The Game class consists of teams, and each Team consists of players. This composition allows for a hierarchical structure to manage games, teams, and players efficiently.



**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.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| **Server Side** | Flexible terminal commands to configure the server, access, or make changes.  Characteristics It is popular in web hosting  Advantages It is upgradeable, it has various options for different web hosting requirements  Disadvantages It is less preferred for web hosting services | The same goes for mac plus more cost-friendly  Characteristics Secured, most preferred.  Advantages Security flaws are caught before they become an issue, it is the most preferred choice for web hosting services  Disadvantages It is more difficult to find applications to support the web hosting required needs. | More software available compared to other OS.  Characteristics It is dominant to the other platforms. Close platform  Advantages High resource requirements, less loading time, high comfortability  Disadvantages easy virus susceptibility, poor tech support | It's better if the server is immobile and can be tracked in a single place. Specifications are better in other devices.  Characteristics More popular, high portability.  Advantages Have a wider reach, better compatibility, cost-effective  Disadvantages It is highly selective to various smart mobile devices Poor security |
| **Client Side** | Moderate expertise and time required. Cost similar to windows. What is required of the application development process to ensure the application is compatible with all web browser platforms and mobile devices? | Maximum expertise and time required. Minimum cost. What is required of the application development process to ensure the application is compatible with all web browser platforms and mobile devices? | Minimum expertise and time required. Cost similar to mac. What is required of the application development process to ensure the application is compatible with all web browser platforms and mobile devices? | Provides flexibility to clients or even developers to see updates at any place. Slightly more difficult to implement than other devices. |
| **Development Tools** | When running languages on macs we can run swift the more popular option. While mixing in nice tools like notepad++. Though Macs can run all languages. Languages consist of but not limited to HTML/CSS/JavaScript while supporting libraries to support the frontend and general-purpose languages. These can be Java, Python, PHP, and Ruby. | Linux can work with visual studio, eclipse, along with notepad++ for a nice and easy-to-use tool. Along with many more languages and tools. Languages consist of but not limited to HTML/CSS/JavaScript while supporting libraries to support the frontend and general-purpose languages. These can be Java, Python, PHP, and Ruby. | Easier to use than Linux but can run the same as it. So visual studio, eclipse to name a few of the many languages. And with multiple tools notepad++ is a simple to use the tool. Languages consist of but not limited to HTML/CSS/JavaScript while supporting libraries to support the frontend and general-purpose languages. These can be Java, Python, PHP, and Ruby. | You can create countless apps using android and swift. Both languages and software can be run on all three machines. Languages consist of but not limited to HTML/CSS/JavaScript while supporting libraries to support the frontend and general-purpose languages. These can be Java, Python, PHP, and Ruby. |

**Recommendations**

Recommendations:

1. Operating Platform:

For expanding "Draw It or Lose It" to other computing environments, I recommend utilizing a cloud-based server platform. Cloud platforms like Amazon Web Services (AWS) or Microsoft Azure offer the scalability, flexibility, and reliability required to support the game across different operating systems.

2. Operating Systems Architectures:

The chosen cloud platform typically employs a virtualized architecture. Virtual machines (VMs) or containers can be used to encapsulate the game application, making it easier to deploy and manage across various operating systems. This approach allows for abstraction from the underlying hardware and offers consistent performance across different environments.

3. Storage Management:

For the recommended cloud-based server platform, an object storage system can be utilized to manage game-related data. Amazon S3 or Azure Blob Storage are suitable options. These systems provide scalable and durable storage, allowing easy access to multimedia assets, user-generated content, and other game data across different platforms.

4. Memory Management:

The recommended operating platform's memory management techniques involve efficient allocation and deallocation of memory resources to the "Draw It or Lose It" software. This includes optimizing memory usage, managing memory leaks, and employing garbage collection mechanisms. Cloud platforms often provide tools for monitoring memory usage and performance, ensuring optimal resource utilization.

5. Distributed Systems and Networks:

To enable communication between different platforms, a microservices architecture can be employed. This involves breaking down the game into smaller, interconnected services that can communicate via APIs. RESTful or gRPC APIs can facilitate seamless interaction between game components across different platforms. For network connectivity, a combination of public and private networks can be utilized, ensuring secure data transmission.

Dependencies between components in distributed systems should be managed by implementing failover mechanisms and load balancing. Redundancy and replication of critical components can help mitigate connectivity issues and outages.

6. Security:

Ensuring user information protection and security across platforms is crucial. The chosen cloud platform should offer robust security features, including data encryption at rest and in transit, identity and access management, and multi-factor authentication. User data should be stored securely in databases with proper access controls.

User interactions with the game should be secured through encryption protocols like HTTPS. Regular security assessments, penetration testing, and timely security patches are essential to safeguard against potential vulnerabilities.