

CTS - Creative Tech Solutions

**CS 230 Project Software Design Template**

Version 1.0

**Table of Contents**

**Document Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| 1.0 | 1/19/2023 | Isa Ali | Initial Commit |
| 1.1 | 2/24/2023 | Isa Ali | Final Additions, completed Recommendations section |

**Executive Summary**

The Gaming Room is a new web based game that I will be developing under the company Creative Technology Solutions. They want to expand the game from mobile to a web based game. My objective will be to help set up a design document that will address the issues they are facing. This will allow them to develop a web app that will bring in more players to their platform.

**Requirements**

- Game will need one or more teams

- Each team will have multiple players

- All game and team names must be unique

- Only one instance of a game can exist in the memory, so we must use a singleton.

**Design Constraints**

- The web based version must be the same as the android version

- The software must make use of a singleton

- We will have to work with the web dev team to make the game as similar as possible to the mobile version

**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**

<Describe the UML class diagram provided below. Explain how the classes relate to each other. Identify any object-oriented programming principles that are demonstrated in the diagram and how they are used to fulfill the software requirements efficiently.>



The first thing to examine is the Program driver class that is responsible for starting up the application. This uses a singleton design pattern so only one instance of GameService can run.

The getInstance() method is responsible for starting the game, it checks if there is already an instance of GameService running and only allows it to start if it is not already there.

After this, it calls addGame() which is responsible for adding new games to the list of games object. This only allows unique game names to be added due to its use of the iterator design pattern.

Next the addTeam() method is called which adds a player to the Player list of teams.

Afterwards the player is also added to a player list through the addPlayer method.

All three of these methods, Game, Team, and Player are sub-classes of the parent "Entity". Entity stores two attributes, "Name" and "Id".

The object oriented design techniques used in the diagram above are as follows. Polymorphism and inheratance are seen in the initial part, where they extend the Entity class to the 3 other classes. There is also abstraction and encapsulation in the team adding process.

**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 OS X is a stable platform for hosting servers. As it is also based on the BSD system with linux, it is able to to handle server hosting. Still, most support for mac OS servers are depricated and out of use. Mac's are no longer advertised as server hosting machines and Apple has recently been removing support for servers. | Linux is a free to use OS that is very popular. Linux allows you to easily install software and scripts for your customers.  They also are easier to set up user permissions, management, and stability in uptime for the operator.  Many software licenses on linux are also open source reducing cost.  Disadvantages of Linux hosting is the need for advanced skills in linux servers, as users need extensive training to properly utilize it which can be costly.  Linux servers can also be more vulnerable from viruses and malware. | Windows servers use Microsoft's inany.NET language, which provides good support for server developent. Windows also has many enterprise applications which organizations can subscribe to for improved features.  The main disadvantage is Microsoft's requirement to use a server version of windows for servers that requires an expensive licensing fee. Most products for windows servers are also closed source, so they cost expensive licensing fees. | Mobile devices are not typically made for hosting servers. The android and iOS operating systems provide few features for server hosting.  They also typically have weak processors and low RAM which make them unsuitable for server hosting. |
| **Client Side** | You first need to install XCode, the propriotary MacOS editor. You will also need to be on the latest version of MacOS.  The primary language in use on apple devices is Swift which is a C-Like language.  It will require a lot of experience with development to work on MacOS software, as it is a low level language and a unique language that will require you to learn form scratch.  MacOS allows multiple user accounts to be created allowing for user management. | Linux allows users to program in any language of their choice, as it is able to execute any source code.  Common languages to write linux apps are C++, Python, and Java.  The time to develop should be lower as the languages are more familiar.  Expertise is at a medium level, as the developer will need to be familiar with the linux system. | Windows development is typically low cost as licensing fees are cheap and anyone can develop applications with C# and .NET on the platform. Many free editors such as VS Code are available for windows.  Windows is also able to support multiple users on their platform easily, which is a core feature.  Development times should not be too long as windows applications have a low barrier to entry.  Experience needed is high as windows applications can be complicated. | Mobile development uses the Android IDE which makes use of Java or Kotlin to make their applications. This requires retraining as mobile is a different platform from windows.  iOS development also uses Swift so some skill can carry over from MacOS development.  Development times are long on mobile as apps must be developed for each platform individually. |
| **Development Tools** | The primary tool used to develop MacOS is the XCode editor. This makes use of Apple's custom language called Swift. This is a language that originates from C and Java. | Linux makes use of many languages including Java, C++, and Python.  Many IDE's are available on Linux, including Vim, Atom, IntelliJ, and Visual Studio. | Windows makes use of the C# and .NET language to develop in the front end and backend for applications.  Microsoft provides the VS Code editor for native windows development. This is a powerful editor that allows users to create applications from scratch for the Windows operating system. | Mobile devices are Programmed with Java, Kotlin, or Swift.  To develop on Android you need the IntelliJ Android Studio editor which is build by JetBrains. This allows you to handle the UI and the backend.  To develop on iOS you need to use the xCode editor. This is provided by Apple and allows you to handle all parts of developing and publishing iOS apps. |

**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**: The best operating platform for The Gaming Room to choose for running the server would be Linux. This choice is good because Linux provides a low-cost operating system to build their server with, since most versions of it are free. Linux also has the admin capabilities to manage a large server. It also has built in security and add on software that can ensure it functions securely. It has dedicated versions of the operating system designed to be used with servers which will benefit them here.

Using Linux also provides The Gaming Room with the ability to support many of the most popular backend languages, like Node, .NET, and Java. As the backend is compatible with any front end they decide to use for the game it will be possible to run this backend on this server.

* **Operating Systems Architectures**:

The ideal architecture for the game would be a backend server that is cloud hosted and stores all the game data like images and player information which can be requested as the player progresses through the game, reducing the load on the servers at any one point. Asynchronous loading of the data can happen while the player is on the current round, the data can load in the background for the next round such as the images and text needed. This can provide for a seamless experience to the player. Using this method of front-end client-based loading will allow the game to run smoother and maintain peak performance without freezing.

The game can be developed as a desktop and mobile app, which will allow it to store basic information on the device reducing the amount that needs to be loaded. This can be programmed in Java or Swift for android and iOS, and on PC it can use C#.

* **Storage Management**: The ideal choice for the gaming room would be to use an HDD based server which will allow for them to utilize larger storage space and increased reliability without compromising on performance for the game.

On the client side it will require a minimal amount of space to store the basic game application and related player account data. The images will be loaded from the server to the RAM as the game is running and then be deleted after the round is over. This will ensure that the client side is not utilizing an excessive amount of storage.

* **Memory Management**:

An ideal client-side memory management method would be to cache the images through the app using a system provided memory management method, like page caching. This can then be managed by the platform and freed up by the garbage collector when needed. Once the round is complete and the player has progressed, the images can be freed up from the memory and new images can be loaded in for future rounds. Images will need to be able to load in quickly for the user to not be affected by internet speeds and device performance, so it should have a high priority in the storage.

* **Distributed Systems and Networks**:

The ideal distributed system to use here would be a cloud hosted server architecture that would allow for The Gaming Room to use the minimal amount of server performance that they need at any one point. If active users for the game spikes, they will be able to request greater server performance and increase the player capacity quickly.

Since they will implement a RESTful API backend, the frontend can be compatible with any platform that the game will operate on as the backend can load the data and communicate if the proper end points are accessed.

* **Security**: <Security is a must-have for the client. Explain how to protect user information on and between various platforms. Consider the user protection and security capabilities of the recommended operating platform.>

The Gaming Room should employ anti-virus and anti-malware software on their servers to reduce any chances of unauthorized code running and affecting their game data. They should also lower minimum account authorization levels to on a as-needed basis. This means all game users should only have permissions to access the basic information needed to play the game, like the images and account information. Admins should only be the game developers and the account should be secured with 2 factor authentication.

They should also use the latest industry standards for server encryption and protection, like TLS certifications and employ the use of a properly configured firewall to prevent outside attacks. They should also include a DDoS protection software, which will prevent client attacks from shutting the server down. A popular option is Cloudflare. All API endpoints should also only send encrypted data protected with a key which will prevent any man-in-the-middle attacks from intercepting the data.