THE GAMING ROOM

<Win, Lose or Draw>
CS 230 Project Software Design Template
Version 1.0
Table of Contents

Document Revision History

Version	Date	Author	Comments
1.0	<05/22/21>	<Matt Zindler>	<initial documentation>

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 program we are looking to create together has several key functions to keep in mind to a successful completion. The game must be multiplayer, and have online capabilities to use the web based services. The game must also be able to keep track of players and teams, making sure there are no duplicates in the process. The game also must only have one instance of each game running at one time to cut down on errors and hardware use. This can be done through the use of singletons for names and teams, which are only created once, and refer to the first instance whenever the code would want to create another version.>

Design Constraints

<The main constraints for web based environments would be networking and server issues. The more complex the program is, it will become even slower than normal due to communications with the web host. Communications between the user and the server the application is hosted on adds additional time for data transfer for each communication necessary. This adds an obstacle for where you would like to store your data. There are also issues with web traffic. If there are too many players the servers can be overloaded and cause a shutdown.>

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

<There are a few main classes that will be utilized by the program. The main class is where the idea of the code is written, utilizing functions from all the other classes. The singleton tester tests each class to see if it only uses one instance of whatever it creates, saving hardware usage and helping remove errors. Game service, game, team, and player classes all keep track of their respective info in an out of sight out of mind type way (encapsulation/abstraction) Game service keeps track of the game library for the user, making a list of all games. Game helps game services keep track of games by bundling relevant info together. The team class helps the Game class by organizing players on each team. And the player class helps the team class by holding all the info for each player, such as user names. All of these classes work together as sorts of building blocks by keeping info together and passing on only what they need to (inheritance/polymorphism).>

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	Works well with server commands. Easy to customize with medium cost.	Great server commands and customizable similar to Mac, but costs less.	Works with the most common applications and software. Very compatible with a medium cost.	The server works well if it's centralized, and can have flexible pricing.
Client Side	Moderate developer expertise to learn the systems. Average cost and time.	Steep learning curve for developers and larger time requirement, but has lower cost with an experienced dev.	Lower learning curve than Mac, as well as an abundance of developers. Average cost and lowe time.	Flexible in cost and time, as well as how much expertise you are looking for.
Development Tools	Mac OS uses HTML for web based uses, Swift, Java, and Python as languages for developers. PyCharm and Xcode are popular IDEs.	Linux is able to take advantage of most of the regular IDEs other OS use: PyCharm, eclipse etc… Other specific IDEs include programs like NetBeans and Geany.	Windows uses many languages, Java, Python, C++. Windows has access to a large variety of IDEs with these languages: Eclipse, PyCharm and others. Since there is a large variety there is also flexible pricing depending on what you need.	Java and Python are popular languages for mobile, as well as Swift for IOS. Java and python can use their IDEs, as well as other IDEs such as AIDE for mobile specific OS's.

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:

<Windows is the most widely used OS, meaning that it can be expanded fairly easily, and many developers are familiar with the OS. Mobile devices also have a wide variety of ways to be expanded, and may be necessary if the mobile playbase were to pick up. I would recommend Windows as the default OS.>

- Operating Systems Architectures:

<Windows employs x64 or x86 for their general system architecture. It is also widely popular as well as Windows. The x86 architecture runs 32 bit systems, while the x64 system runs 64 bit. X64 is able to utilize more memory at a given time than its 32 bit counterpart. Therefore it is recommended as it allows the user to have more processes running at the same time, or allow the game to utilize more memory for more functionality.>

- Storage Management:

<SSD's are a great choice for storage management. They are the faster cousins of HDD's, being able to store, write, and retrieve data faster than an HDD. HDD's could be considered though, as they will be cheaper but have worse performance.>

- Memory Management:

<Windows is recommended for memory management. Windows uses virtual and physical memory to handle tasks. With virtual memory, the OS holds onto processes until physical memory opens up to run the program. The 64 bit system is able to have multiple processes running at the same time.>

- Distributed Systems and Networks:

<The network and servers will need to be able to handle multiple players from different locations. Highly optimized networks that are able to handle the required traffic will be a requirement so the game does not go offline due to player activity. Additionally, having extra servers (not just in one location) can help in the case of a set of servers going down. If there are multiple locations, then the risk of having everything go down is reduced.>

- Security:

<Security is important for end user security. Keeping their accounts safe, and also keeping the servers safe from malicious attacks is a top priority. As for keeping their accounts safe, multi factor authorization or an access matrix can be implemented to limit the exposure of user accounts, or actions taken by a malicious individual.>