The Gaming Room

CS 230 Project Software Design Template

Version 3.2

Table of Contents

Document Revision History

Version	Date	Author	Comments
1.0	5/28/23	Tatiana Gabel	Created Executive summary, updated requirements, and design constraints
2.0	5/30/23	Tatiana Gabel	Added Domain model
2.1	6/2/23	Tatiana Gabel	Added UML diagram
3.0	6/16/23	Tatiana Gabel	First draft of OS requirements client and server side
3.1	6/20/23	Tatiana Gabel	Updated OS requirements table (Evaluation)
3.2	6/24/23	Tatiana Gabel	Added OS recommendations

Executive Summary

The gaming Room wants to develop a web-based game that will be used on multiple platforms. They currently have an android-based version of the game. The development of this application needs to be streamlined. The game will include an option to play with a single team or multiple teams. It also must prevent users from creating duplicate team names.

Requirements

- A game will have the ability to have one or more teams involved.
- Each team will have multiple players assigned to it.
- Game and team names must be unique to allow users to check whether a name is in use when choosing a team name.
- Only one instance of the game can exist in memory at any given time. This can be accomplished by creating unique identifiers for each instance of a game, team, or player.

Design Constraints

- Consistent Gameplay for both a single team or more than one team, this will require the development team to implement the Game class in a way that supports multiple teams, such as using a list of teams rather than a single team attribute.
- Players must be grouped within a team, again this will require the development team to add an attribute to team class will which players are on that team, additionally players may also need to contain an attribute with their team.
- Players within the same team receive the same information, this will require a way for the game class to communicate with all the players on a team, this could be done using a *teams* class.
- Check for game and team names to prevent duplicates. The implications of this is likely a list of team names will need to be maintained, in order to check for repeated names before instantiation. The Game object should instead adopt a singleton pattern, to prevent duplicate instantiations.

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 is used as an outline for its child classes: Game, team and Player. It contains relevant information we will use, such as id, name, and a toString() method. Game, team and Player will inherit these attributes from Entity so we can give them an id and name. These classes may overload the toString() method to display the relevant class type. This will make our code easier to understand and more efficient when programming classes as less information will be repeated. Additionally, it still allows for flexibility in the child class definitions.

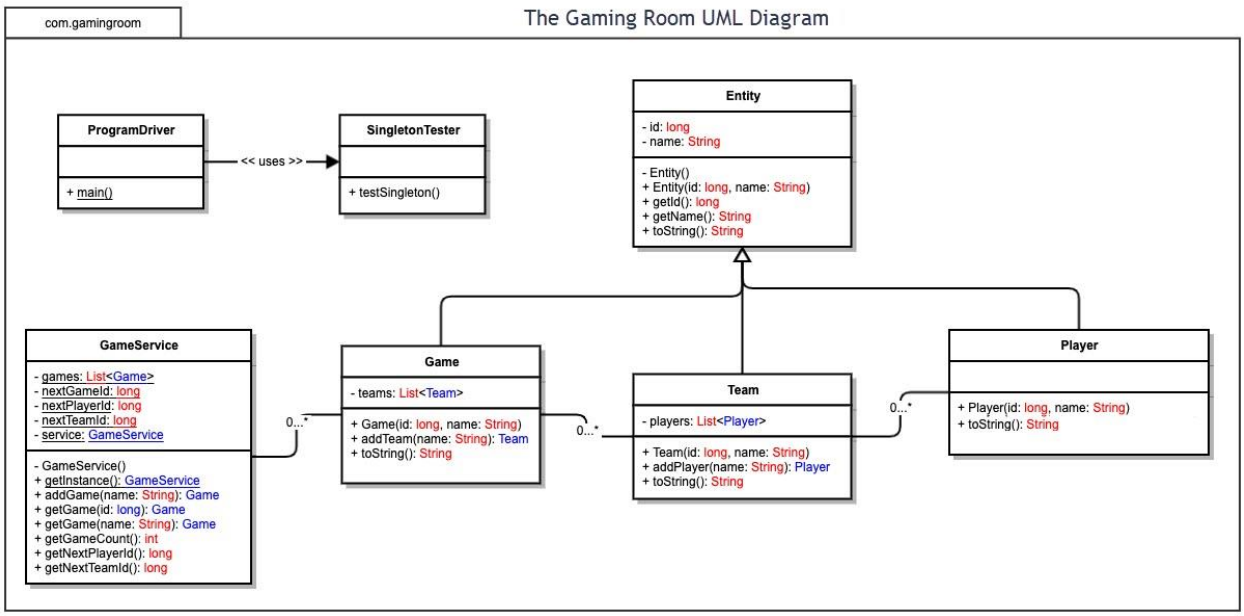
In addition to the attributes listed above, the Game class also includes a list of teams, a constructor, as well as the ability to add teams. As we can see on the UML diagram, the game can have none to many teams.

Similarly, the Team class includes a list of players, a constructor, and a method to add players to the team. This is also shown on the diagram as a team can have none to many players.

Finally, the Player class contains a constructor.

The Game service object implements the singleton pattern, which restricts it from being instantiated multiple times. It contains a list of Games, the Id of the next game, the id of the next player, and id of the next team. It also allows the user to add a game, get game, view team or player id, and get the number of games. From the diagram, the Game Service class does not inherit from another class in the diagram. It may have none to many Games. Implementing the check internally is more efficient than implementing the check externally, which would require the program driver to contain a GameService attribute.

The program driver contains the main method, which is where program execution will start. The SingletonTester class will test if there only one instance of the GameService class.



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	A Mac OS is a very limiting server choice as it is primarily used to develop Mac and Ios apps. However, it does offer a lot more flexibility when it comes to setting permissions and changing resources, as those changes can be synchronized across different apple products. Using a mac OS would require an increased server cost, as the MacOS Server run by apple has been discontinued, which means that only a third-party or cross-platform server is available to use.	Linux is regarded as a strong option for hosting a website. Some advantages are stronger security, and support for other development tools, something that was a problem with macOS and windows servers. Hosting on Linux generally also costs less money than hosting on windows server. However, there are several linux distributions, which vary widely in support, flexibility and customization. Which would require more money to complete the necessary research and hire some specialized in linux servers.	Windows servers do not use the same OS that personal computers do. Katie Horne explains that Hosting the server on windows will cost more, because of the licensing fees. There is also specialized software requirements, as many windows servers use an SQL database and the .NET framework.	A mobile app will depend on an API to access a server. Our application requires use of the server to log in users, communicate between players, among other things so this would be essential. Additionally, many mobile devices require push notifications to be done via server communications. Because of how much needs to be done via the server-side, as well as a growing amount of users. The cost will be larger than the server cost for personal computers.

Client Side	One key advantage, is the ability to develop for both the MacOS and other apple products at the same time. So if this application was prepared to launch on the MacOS through Xcode, it could also be launched on other Apple devices. However, that is also limitation, since not all hardware uses the MacOS. There are dedicated Mac developers, and because of the benefit of developing for both Mac and Ios at the same time the cost is still reasonable. There are fewer ways to test the web app without a mac device, although xcode does provide a MacOs simulator. The browser that is unique to Mac and Ios devices is Safari. Some versions of Safari require CSS vendor prefixes(A way for browsers to locate features specific to them), so the website is consistent across all versions.	As we are developing an application that runs almost completely within the browser, the Linux operating system provides sufficient support. Linux-only browsers are less popular than safari or internet explorer, so no additional design considerations are necessary. Linux also has good security, and most customization. One downside is that Linux developments may need specialization based on which distribution and version of linux you plan to use. However, Linux developers are very common, which means that it also would have a lower development cost and shorten development time.	Windows is very commonly used, so it would apply to many clients. Similarly, windows offer a large amount of development tools to simulate the Operating system. Hence, more developers are likely to have experience with the windows OS, which lowers the cost and time requirements for developing the application. The unique clients we might encounter on a windows computer are users using internet explorer or Microsoft Edge as their internet browser. Both of these are associated with being older, so making sure that our web application is compatible with previous versions of these browsers, while still being safe and secure for other users is essential. This might include, supporting longer response times or decreasing the amount of information sent.	A good solution for creating mobile apps with a lower budget is to design a cross-plaform application, or a progressive web app, as the Onix-systems blog explains. (Sheremetov 2023) This means that the majority of development is in the form of web development, but the layout and UI is changed to work with a mobile device. However, it will still require experts in more than one OS, because IOS and Android applications use different development tools. Which nearly doubles the cost of creating the application(because in many ways 2 applications are being created). One way this can be leveraged in our favor is by using the MacOS to develop for IOS devices. Or similarly using the Linux system to develop for android devices. However, another disadvantage is hosting an application on both the App Store and Google Play store means paying the required fees for both of these services.

Development Tools	We can still use Java when programming for Mac applications. However, Apple also provides some other free apps such as Xcode, which allows users to test their applications with a simulated macOS and design UI specifically for Apple operating systems. Swift is the language commonly used, as it not only works with the MacOS, but also for the ipad and iphone operating systems.(Team 2022) Additionally, in order to publish the app on the apple App Store there is an required fee that includes testing tools through the Apple developer Program (Team 2022)	Linux applications can be created in a variety of languages, which are then compiled and installed as packages for the linux shells to use. We plan to use CSS, Html, and the REST API through Java, so Eclipse or Visual studio code would be good options. Both are free software, and Linux does not include any licensing costs. For deployment, GitLab CI/CD is primarily designed for software development. Alternatively, many deployment applications used for windows can also be used for Linux, such as Azure or AWS.	Windows applications also are created in a variety of languages. So the same IDEs that are used with linux can be used here. Some Windows deployment applications and IDEs require licensing fees and may also require specialized tools such as the .net framework. Microsoft Azure is one option for hosting and building web applications, especially when a growth of users is expected. Alternatively Amazon web services can also be used for web development. Both of these services require a fee, but AWS allows for more freedom in cost.	For mobile development we have different tools for different device operating systems. For IOS devices we can use Xcode with objective-c and Swift as the programming languages, like we would in the development of mac applications. For android applications, there is Android studio, which includes code samples and debugging tools to developers. There are also many existing SDKs to help applications meet android requirements. Similar to how IOS and mac apps can be made together, the google play store provides deployment to android devices as well as ChromeOS and Windows. If we want to stay consistent with the other operating systems development process, then visual studio code can also be used to development mobile application when combined with Xamarin.

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**: <Recommend an appropriate operating platform that will allow The Gaming Room to expand Draw It or Lose It to other computing environments.>

2. **Operating Systems Architectures**: <Describe the details of the chosen operating platform architectures.>

3. **Storage Management**: <Identify an appropriate storage management system to be used with the recommended operating platform.>

4. **Memory Management**: <Explain how the recommended operating platform uses memory management techniques for the Draw It or Lose It software.>

5. **Distributed Systems and Networks**: <Knowing that the client would like Draw It or Lose It to communicate between various platforms, explain how this may be accomplished with distributed software and the network that connects the devices. Consider the dependencies between the components within the distributed systems and networks (connectivity, outages, and so on).>

6. **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.>

Sources:

Butler, R. (2022, February 14). Which OS is Best Suited for Your Development. BairesDev.

Develop Android games. (2023). Retrieved June 12, 2023, from Android Developers website: https://developer.android.com/games

Gawin, M. (2021, April 23). Best Android App Development Tools & Software. INVO Blog. Retrieved June 12, 2023, from INVO Blog website: https://invotech.co/blog/best-android-app-development-tools-software/

Horne, K. (2021, May 4). Which Operating System Should You Choose for Your Web Hosting Server? Digital.com; Digital.com. https://digital.com/best-web-hosting/operating-systems/

O'Leary, R.(2021, May 5). Is Vendor Prefixing Dead?. https://css-tricks.com/is-vendor-prefixing-dead/.

Sheremetov, D. (2023, April 11). How Much Does It Cost to Make a Mobile App? Onix-Systems.com; Onix. https://onix-systems.com/blog/how-much-does-mobile-app-development-cost

Silberschatz, B.,Galvin P.,Gagne G. (2008) Operating System Concepts. 8th Edition.

Team, S. (2022, May 16). Develop Apple Apps: Learn Mac & iOS App Development, Xcode & Swift — SitePoint. Sitepoint.com; SitePoint. https://www.sitepoint.com/develop-apple-apps/