

The Game Room web based game

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template**](#_l6ti7uoag22u)1

[**Table of Contents**](#_30j0zll)2

[**Document Revision History**](#_grjogdjh5fi8)2

[**Executive Summary**](#_sbfa50wo7nsh)3

[**Design Constraints**](#_2et92p0)3

[**System Architecture View**](#_ilbxbyevv6b6)3

[**Domain Model**](#_8h2ehzxfam4o)3

[**Evaluation**](#_2o15spng8stw)3

[**Recommendations**](#_m8aleynsvzvc)5

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 11/11/2021 | Dale Ayers | In this version we are adding the Entity class and revising the rest of the code to integrate it. |

**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](#_sbfa50wo7nsh)

We need to build a web-based game that serves servs multiple platforms based on the game Draw it or Lose it. We need to set up the environment, streamline development.

## [Design Constraints](#_2et92p0)

The design constraints are the game needs to allow one or more teams, each team will be able to have more than one player, each team name should be unique and checked against existing team names, only one instance of the game can exist in the memory at one time

## [System Architecture View](#_ilbxbyevv6b6)

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](#_8h2ehzxfam4o)

I know that this is a singleton pattern from the singleton tester meaning that there can only be one instance of the GameService class. The Entity class is the Parent Class and the Game, Team and Player classes are the child classes which all inherit Variables and Methods from the Entity class. The GameService, Game, Team and Player classes all have an associative relationship, this can best be explained by one GameService can have zero or multiple Games, one Game can have zero or multiple Teams, one team can have zero or multiple Players.

**"The Gaming Room UML diagram. The top of the diagram is labeled as com dot gamingroom. Test boxes are placed in two layers. The first layer has three text boxes and the second layer has four of them. In the first layer, the 'ProgramDriver' textbox points to 'SingletonTester' textbox. The 'ProgramDriver' textbox contains the text 'asterisk main round brackets.' The 'SingletonTester' textbox contains the text 'asterisk testSingleton round brackets.' The arrow between these two text boxes are labeled 'open two angle brackets uses close two angle brackets'. In the second layer, there are 'GameService', 'Game', 'Team', and 'Player' text boxes. The 'GameService' textbox has texts arranged in two layers. The first layer contains games colon List open angle bracket Game close angle bracket, nextGamesId colon long, nextPlayer Id colon long, nextTeamId colon long, and service colon GameService. The second layer contains GameService round brackets, getinstance round brackets colon GameService, addGame open parenthesis name colon String close parenthesis colon Game, getGame open parenthesis id colon long close open parenthesis colon Game, getGame open open parenthesis name colon String close open parenthesis colon Game, getGameCount round brackets colon int, getNextPlayerID round brackets colon long, and getNextTeamId round brackets colon long. The 'GameService' box is connected with the 'Game' textbox with a line labeled 'zero dot dt dot asterisk'.  The 'Game' textbox also contains text in two layers. The first layers contains the text teams colon List open angle bracket Team close angle bracket. The second layer has Game open round bracket id colon long comma name colon String close parenthesis, addTeam open parenthesis name colon String close parenthesis Team, toString round brackets colon String. The 'Game' textbox is connected with the 'Team' textbox with a line labeled 'zero dot dt dot asterisk'. The 'Team' textbox also contains text in two layers. The first layers contains the text players colon List open angle bracket Player close angle bracket. The second layer has Team open parenthesis id colon long comma name colon String close parenthesis, addPlayer open parenthesis name colon String close parenthesis colon Player, and toString round brackets colon String. The 'Team' textbox is connected with the 'Player' textbox with a line labeled 'zero dot dt dot asterisk'. It contains the text Player open parenthesis id colon long comma name colon String close parenthesis and toString round brackets colon String. The 'Game', the 'Team, and the 'Player' boxes point to the 'Entity' textbox in first layer. The 'Entity' textbox contains text in two layers. The first layer has the text id colon long and name colon String. The second layer has Entity round brackets, Entity open parenthesis id colon long comma name colon String close parenthesis, getId round brackets colon long, getName round brackets colon String, toString round brackets colon String.**

## [Evaluation](#_2o15spng8stw)

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 has almost no viruses, mac is also less prone to hardware and software crashes because of their mac-apple relationship | Linux is the software preferred on most web development services and renowned for its flexibility | Windows has a wide variety of choices, compatibility almost every application driver or game works on Windows | One big disadvantage is not knowing if you’re going to have access to the internet depending on the service youre getting. An advantage would be ability to access anywhere there is sufficient cell service. |
| **Client Side** | When it comes to mac the cost may be more because the hardware is specialized where as windows is much more popular. | Linux is not standardized so running into problems that would require a “Linux” might be a problem because there is no one “Linux”. Linux requires much more hands on expertise than other options. | Cost is similar to Mac but the advantage with Windows is the cross platform compatibility and no need for specialization the way you have to with mac. The gaming submarket around windows is much bigger than both Linux and Mac | Ability to access updates from anywhere but harder to implement than on a desktop. |
| **Development Tools** | If you sufficiently equip your mac you can use parallels and other virtualization application to provide multiple operating systems on mac. Macs can run all languages | Wide variety of devices that support Linux. No matter the office app, media production app or coding tool, Linux has versions available usually with no fee or subscription, its just going to require more expertise to run. 0 | Windows incorporates “out of the box” with Active Directory-based corporate servers. Windows tools are widely available affordable and standardized. | Ability to create apps on android and swift both languages are supported on android, Linux and mac. |

## 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**: When it comes to recommending an operating system, we should look at growth and community. Mac has some exciting features that will allow you to run different features of different operating systems on their operating system, and Linux has an open-source community that is commendable for people interested in that topic. Still, when gaming and creating a large community around your game, the Windows operating system server is second to none. Windows owns 75% of the market in the desktop and laptop market. Steam, the digital video game distribution platform, publishes its "Hardware & Software Survey" statistics, and the numbers are even more staggering when it comes to gaming. Of the people using Steam as of November 2021, 96.14% were using Windows operating system. (Steam 2021) The choice is clear when it comes to gaming; Windows operating system is king.
2. **Operating Systems Architectures**: When it comes to recommending an operating system, we should look at growth and community. Mac has some exciting features that will allow you to run different features of different operating systems on their operating system, and Linux has an open-source community that is commendable for people interested in that topic. Still, when gaming and creating a large community around your game, the Windows operating system server is second to none. Windows owns 75% of the market in the desktop and laptop market. Steam, the digital video game distribution platform, publishes its "Hardware & Software Survey" statistics, and the numbers are even more staggering when it comes to gaming. Of the people using Steam as of November 2021, 96.14% were using Windows operating system. (Steam 2021) The choice is clear when it comes to gaming; Windows operating system is king.
3. **Storage Management**: With serverless architecture, local storage will not be a significant concern because a lot of our storage will be on the cloud. It may be appropriate for us to have some local storage, though, and for that, I would recommend that we use Solid State Drives for quicker and more reliable storage of information than a Hard Drive.
4. **Memory Management**: For memory, the consideration is what kind of gaming audience are we catering to? If this were a high-end first-person shooting game, then the amount of memory or RAM required for each user to play the game would be much more. But for this game which is much more turn-based and does not require a massive amount of memory, I don't think memory is something we have to worry about since almost any consumer computer or laptop has enough ram to play this game.
5. **Distributed Systems and Networks**: If we set our game up using serverless architecture, then our game will be a great example of a distributed system. We have the server provided by an outside provider like AWS; then we have the users who will play the game on their computers using their local resources on their computer to run the game while the client and the host are communicating. AWS will be able to provide many different resources so that our server is not just dependent on one machine. Doing this provides benefits such as computation speed and reliability. When your server is not connected to one machine, one machine's failure will not lead to the server failing, increasing reliability.
6. **Security**: When starting to think about building secure software, we should use the principle of least privilege; it dictates that programs, users, and even systems be given just enough privileges to perform their tasks. A great example of this is the analogy of a security guard with a passkey. If the passkey is only allowed access to the public areas that the security guard guards, then if the passkey is stolen and the thief uses it maliciously, the damage would be minimal than if the passkey allowed the thief access to every door in the building.  Simply put, we should use the principle of least privilege from the ground up when developing our program.  Keeping in mind the principle of least privilege when adding multiple users onto a system entails creating a user account for every user and only allowing them as many privileges as the user needs; allowing more privileges than required for convenience goes against the principle of least privilege. As for Windows, security features include a lot of anti-virus and antimalware protection built-in. Firewall protection and account protection are some of the best features included in Windows as well

References

Ashrafi, M. (2020). File systems: Mac vs. Windows vs. Linux. Retrieved from https://macinfo.us/file-systems-mac-vs-windows-vs-linux/

Denisebmsft. (n.d.). Windows operating system security. Retrieved from https://docs.microsoft.com/en-us/windows/security/operating-system

Gewirtz, D. (2019, April 8). Windows, Mac, or Linux? We compare the pros and cons of these computing platforms. Retrieved from https://www.zdnet.com/article/windows-mac-or-linux-we-compare-the-pros-and-cons-of-these-computing-platforms/

Roomie, M. (2021, February 24). 5 advantages and disadvantages of operating system | Limitations & benefits of operating system. Retrieved from https://www.hitechwhizz.com/2021/02/5-advantages-and-disadvantages-limitations-benefits-of-operating-system.html

Stevewhims. (2019, August 23). The model for distributed systems. Retrieved from https://docs.microsoft.com/en-us/windows/win32/rpc/the-model-for-distributed-systems

Windows. (2021, April 5). What is the architecture of Windows operating system? Retrieved from https://ostoday.org/other/what-is-the-architecture-of-windows-operating-system.html

Bashir, F. (2019, July 13). What is Serverless architecture? What are its pros and cons? Retrieved from https://www.freecodecamp.org/news/what-is-serverless-architecture-what-are-its-pros-and-cons/

Oreilly. (2009). Operating system concepts, 8th edition. Retrieved from https://learning.oreilly.com/library/view/operating-system-concepts/9780470128725/silb\_9780470128725\_oeb\_c16\_r1.html#h2

Steam. (2021, November). Steam hardware & software survey. Retrieved from https://store.steampowered.com/hwsurvey