

The Gaming Room

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template**](#_30j0zll)1

[**Table of Contents**](#_1fob9te)2

[**Document Revision History**](#_2et92p0)2

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

[**Design Constraints**](#_3dy6vkm)3

[**System Architecture View**](#_1t3h5sf)3

[**Domain Model**](#_4d34og8)3

[**Evaluation**](#_2s8eyo1)3

[**Recommendations**](#_3rdcrjn)5

## [Document Revision History](#_2et92p0)

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| 1.0 | 01/31/21 | Mitchell Lynds | Created class “Entity”, completed classes “Team” and “Player” |
| 1.1 | 02/14/21 | Mitchell Lynds | Completed Evaluation section of Design doc. |

**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](#_tyjcwt)

The Gaming Room project requires that a single game service manage multiple games each with multiple teams and players. To manage these requirements we propose to break the project up into several classes: GameService, Game, Team, and Player. We will use inheritance for the Game, Team and Player classes to reduce redundant code and improve maintainability. We will use appropriate design patterns to ensure proper function of the game service.

## [Design Constraints](#_3dy6vkm)

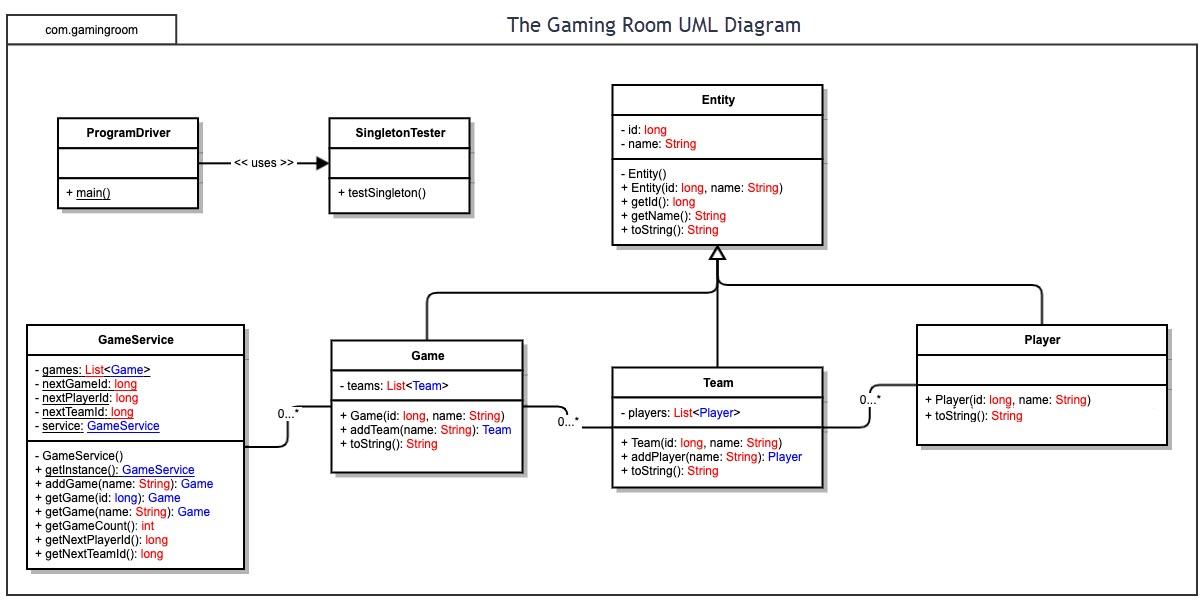
Being in a web-based distributed environment means that the single, central game service will need to be able to host all of the games running concurrently and be able to add and remove games as they begin and end. Since all of the games will be centrally hosted it is important that the system be able to handle many requests simultaneously.

## [System Architecture View](#_1t3h5sf)

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](#_4d34og8)

The program driver will use a singleton tester to make sure that GameService is designed in accordance with the Singleton pattern. This ensures that there is only one GameService instance. The Game Service will have 0 or more games. Each game will have 0 or more teams and each team will have 0 or more players. Each or the Game, Team, and Player classes will inherit from the abstract class Entity. This will cut down on redundant code. The addGame(), addTeam(), and addPlayer() methods will all utilize Iterator patterns in order to ensure each one is unique.

****

## [Evaluation](#_2s8eyo1)

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** | * Con: Mac hosting hardware is very expensive when compared to Linux * Con: Mac Hosting services also tend to be more expensive than other options. * Pro: Mac Server License is a one time fee * Con: Updating to newer versions will incur costs. * Pro: Access to technical support from Apple | * Pro: Linux is likely the cheapest hosting option because of the lack of license fees. * Pro: Linux hosted servers are generally stable. * Con: Linux servers are less user friendly and will require experienced team members to operate. * Pro: Linux hosting is very popular and widely available. * Con: There are many different versions of Linux with varying degrees of support. | * Pro: Windows servers will be familiar in use. * Con: License costs for Windows Server can be high and are a recurring monthly expense. * Pro: Access to Windows Customer Support. * Pro: Quick simple setup. | * Con: While possible, server hosting for web-based applications on a mobile device is almost unheard of. Because of this it will be difficult to find a sufficient web server. * Pro: There are some available options for web server hosting. * Con: Employees with experience using mobile devices to host web servers will be rare to nonexistent. * Pro: Hardware for testing will be relatively affordable. Most mobile devices cost $1000 or less. * Con: Mobile devices operate on wireless internet connections and may not have the bandwidth necessary to host the desired number of games. * Con: Hardware capability is also limited and likely to reduce the number of possible concurrent games. |
| **Client Side** | * Pro: Ensuring Safari compatibility will cover most Mac users. * Con: Hardware for testing will be expensive. | * Con: True Linux compatibility will require the application to run on some niche browsers. * Pro: Hardware and software for testing will be inexpensive. | * Con: Windows compatibility will require the application to run on several browsers. * Pro: Hardware for testing will be inexpensive. * Con: OS licenses in order to test compatibility have costs. | * Pro: The application already runs effectively on Android. * Con: The current application will not run on iOS and will need to be developed from the ground up for that platform. * Pro: Some employees have experience with development for mobile devices * Con: Storage space is limited. Game assets must not be too large or users will not install the application. * Pro: Services like Google Play and the Apple App store are already in place to distribute the application to devices effectively. |
| **Development Tools** | * JavaScript seems to be the most popular language for web-applications. * Other popular languages for web applications include Python and Java. Most of our employees are already experienced with these languages. * Visual Studio Code is a well regarded IDE for JavaScript development and runs on Mac. * Visual Studio Code is free for commercial use as it is open-source but not all potential IDE’s for this project are. | * JavaScript seems to be the most popular language for web-applications. * Other popular languages for web applications include Python and Java. Most of our employees are already experienced with these languages. * Visual Studio Code is a well regarded IDE for JavaScript development and runs on Linux. * Visual Studio Code is free for commercial use as it is open-source but not all potential IDE’s for this project are. | * JavaScript seems to be the most popular language for web-applications. * Other popular languages for web applications include Python and Java. Most of our employees are already experienced with these languages. * Visual Studio Code is a well regarded IDE for JavaScript development and runs on Windows. * Visual Studio Code is free for commercial use as it is open-source but not all potential IDE’s for this project are. | * Pro: Android applications are generally written in Java, which our team is already working in. * Con: For iOS compatibility the application will likely have to be rewritten in Objective C. * Pro: There are mobile device based IDE’s that should allow for easy modification and testing throughout development. * Pro: Development for mobile can also be done in familiar IDE’s such as Visual Studio and Eclipse. * Con: Employees may have to learn a new IDE like XCode in order to achieve IOS compatibility. |

## 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**:

For this project I would recommend using a Linux server. A Linux server offers the features that will be important for The Gaming Room while saving the project a significant amount of money.

1. **Operating Systems Architectures**:

The Linux server uses a “Kernel” which user programs interact with using system-calls. In Linux the kernel contains the device drivers and interacts with the hardware. One layer above that the operating system provides the different function calls that can be made to the kernel by using the C-library. On the next layer the shell is how the user and applications interact with the system invoking system calls from the library which can be passed to the kernel and executed on the hardware.

1. **Storage Management**:

I would recommend a RAID storage management system for The Gaming Room, which Linux supports. The project does not require a massive amount of storage so mirroring that storage on to several drives should be an affordable way to avoid catastrophic data loss that could cause downtime in the game service or loss of user data.

1. **Memory Management**:

Linux uses several mechanisms to manage the available memory of the system. Primarily, Linux uses swapping to avoid running out of memory. Data in memory that goes unused will be copied to “virtual memory” on the hard drive and transferred back to memory if the system tries to access it again. This way Linux is able to run multiple processes at once. Particularly relevant to The Gaming Room Linux also uses sharing to run multiple processes. When two or more processes store the same data in memory Linux is able to make both processes reference the same location in memory effectively making main memory work as if it is bigger than it actually is. In our application this means that even though there will be 1000 instances of the game running there does not need to be 1000 copies of the game’s code stored in memory.

1. **Distributed Systems and Networks**:

The server will use a communication protocol to communicate between various operating platforms. The game will be accessed from several different types of operating platforms at the same time so it is important that each version adheres to the same communication protocol. In this way the information the server receives will be the same regardless of the operating platform the game is being accessed on. A protocol like REST API sets standards for how requests and responses should be sent between distributed systems on a network so that they can be interpreted correctly.

1. **Security**:

Linux servers are well-known for their high level of security. Linux systems use role-based access control as well as other features to protect data. Linux supports the non-executable (NX) feature of AMD and Intel chips. As a UNIX based multi-user operating system the Linux server will be less vulnerable to viruses because the powers of executable files are very limited. Linux also employs encryption in its security features. It is also important that all communication to and from the server by the web application is authenticated and encrypted.