

<Name-of-Software-Application>

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template** 1](#_Toc115077317)

[**Table of Contents 2**](#_Toc115077318)

[**Document Revision History 2**](#_Toc115077319)

[**Executive Summary 3**](#_Toc115077320)

[**Requirements 3**](#_Toc115077321)

[**Design Constraints 3**](#_Toc115077322)

[**System Architecture View 3**](#_Toc115077323)

[**Domain Model 3**](#_Toc115077324)

[**Evaluation 4**](#_Toc115077325)

[**Recommendations 5**](#_Toc115077326)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 5/21/2023 | Brandon Walters |  |

**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)

A company called the Gaming Room wants to convert their already existing android application into a web-based game that works on multiple different platforms. The solution to this software design problem is to develop a system that encapsulates methods that contain efficient software design patterns.

## Requirements

1. Turn the existing android application into a web-based game in order to gain access to more platforms.
2. The UI must be familiar to individuals who have already used the existing android application.
3. Make sure the game and team names are all unique.

## [Design Constraints](#_2et92p0)

1. Cross platform capability: Use of specific technologies in order to run and test on multiple platforms.
2. Network efficiency: Needs to be well optimized due to it being a web-based game, allowing it to handle UE connectivity issues.
3. Multiple Users: Needs to handle multiple users using it simultaneously.
4. Security: Due to its web-based nature it must have proper security in place in order to prevent potential attack on the application.
5. Scalability: Needs to be able to scale efficiently with an increasing number of users due to its web based nature.

## [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)

In this UML class diagram Game, Team, and Player all inherit from Entity showing a inheritance relationship. Furthermore, they all also have a “is a” relationship with Entity because each class is a Entity. Team and Player, Game and Team, Game service and Game, all show a “has a” relationship meaning that one class has a reference to another class.

**"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** | Stable but not often used for server-side applications. | Great for server applications. It is stable and has a lot of options for customizations. | Good for hosting web-based application but can be less stable than Linux when used heavily. | Not good for server-side applications due to limitations such as battery and processing power. |
| **Client Side** | It uses Safari for its browser which may be trickier to develop on, | Needs to be compatible with open-source browsers like Firefox. | Because it has a huge number of users it must be compatible with multiple browsers like edge and chrome. | Needs to be developed in a way that supports the use of multiple screen sizes and operating systems such as Android and IOS |
| **Development Tools** | Languages/Tools: JavaScript, Node.js,  Python, Ruby  IDE: VS Code is preferred due to its wide range of language compatibility and community support. | Languages/Tools: JavaScript, Node.js, Python, PHP  IDE: VS Code is preferred due to its wide range of language compatibility and community support. | Languages/Tools: JavaScript, Node.js  IDE: VS Code is preferred due to its wide range of language compatibility and community support. | Languages/Tools: JavaScript, CSS, HTML5, React  IDE: VS Code is preferred due to its wide range of language compatibility and community support. |

## 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**:

Linux should be used for server-side due to its reliability and JavaScript/HTML5/CSS should be used for client side as it will allow compatibility with all modern web browsers on all operating systems.

**Operating Systems Architectures**:

Linux has a Monolithic kernel architecture meaning it all works in kernel space allowing it to be more efficient with its resources. HTML5/CCS/JavaScript works on all operating systems.

**Storage Management**:

For a storage management system MongoDB which allows it to be efficient and horizontally scalable.

**Memory Management**:

In Linux the kernel handles memory management. Also, in the browser JavaScript has garbage collection which allows the system to free up resources that are not being used.

**Distributed Systems and Networks**:

Because the game is browser based, HTTP/HTTPS can be used for client-server interactions.

1. **Security**:

All over network data should be encrypted with HTTPS. User data should be stored with hash algorithms.