

Adam Sissoko

# **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 | 07/17/2022 | Adam Sissoko | Edited Entity Class |

**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)

The Gaming Room needs to create a game called ‘Draw It or Lose It’ that will run on multiple platforms. The game is currently only available on android devices. We need to create a web-based version of the app, and make sure that it meets The Gaming Room’s software requirements.

## [Design Constraints](#_2et92p0)

* Game needs to have one or more teams involved.
* Each team needs to have multiple players.
* 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.

## [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)

The Game, Team, and Plater class all inherit information from the entity class. These three classes share common references like ‘name’, and ‘id’, which means Entity is a ‘super class’. GameService has a reference from Game, Game has a reference from Team, and Team has a reference from Player.

**"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** | A Mac OS X server costs around $20, and it comes with flexible terminal commands that allow you to make changes quickly and easily. | Similar to Mac, a Linux server would be very inexpensive. Unfortunately, Linux is quite unpopular, so it may be a challenge finding someone with the experience necessary to run the server. | Windows is the most popular OS, so it will be easy to find someone to run the server. However, compared to Mac and Linux, a Windows server may be costly to implement. | Running a server from a phone will be a challenge, since phones have limited computing power, and they can be lost or misplaced quite easily. |
| **Client Side** | Cost Similar to Windows, since it is not open source. Time will depend on the user, since each user has a different expertise level with Mac OS. | Low cost, since Linux is open-source. Time and expertise required will be quite high, since Linux is not popular, and it will be difficult to find someone who has experience with this platform. | Cost Similar to Windows, since it is not open source. Time will depend on the user, since each user has a different experience level with Windows. | Cost will likely not be an issue. Experience will also not be an issue, since mobile devices usually have simple interfaces. More time will be needed, since there are multiple operating systems for mobile devices. |
| **Development Tools** | Swift is the most popular option used to write applications for Mac OS. | Developers typically use IDEs like Eclipse and Visual Studio to create programs for Linux. | Developers typically use IDEs like Eclipse and Visual Studio to create programs for Windows. Programmers can also uses programs like Notepad++ to make the process faster. | Similarly to MacOS, developers can use Swift to create iOS apps. For Android apps, they can use Android Studio. |

When it comes to development tools, Mac, Windows, and Linux have a fair amount of overlap. Most of the IDEs used with Windows computers also work with MacOS. For example, tools like Eclipse, NetBeans, and Visual Studio work on all three platforms, and they are also free. For Android phones, developers typically stick with Android Studio, since it is designed specifically for Android apps, and it has a large community of users, making it easy to get help with technical issues. On iOS, developers can used Swift, an open source IDE designed by Apple.

Using free development tools allows the team to spend more money on the actual development process, resulting in a better overall outcome for the client. For large-scale projects, and projects that use multiple programming languages, it may be a good idea to have multiple development teams, with each team focusing on a different platform.

For each of the operating systems listed above there are plenty of free, open-source development tools, and the team won’t need to spend any money on licensing fees. On Windows alone, there are well-known and popular IDEs like NetBeans, PyCharm, or Eclipse. Many of these open-source IDEs function just as well as their subscription-based counterparts, and they usually have active communities where users can get technical support and advice.

## 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**: I would encourage The Gaming Room to work with Windows devices, since Windows computers are usually quite inexpensive, and the development team will be able to use a wide range of IDEs and programs to create the software. Linux has a smaller selection of development programs to choose from, and Apple computers tend to be expensive and difficult to repair.
2. **Operating Systems Architectures**: Windows Architecture is modular, and it has two main layers, User mode and Kernel mode. In User mode, programs and systems are limited in the amount of resources they have access to. In Kernel mode, programs have unrestricted access to system memory, and they are also able to access external devices.
3. **Storage Management**: Windows 10 computers come with a feature called ‘Storage Sense’, allowing us to see how much space each file takes up, and manage files on the hard drive easily. Windows also comes equipped with ‘File Explorer’, which allows users to find old files easily and delete them to create more room.
4. **Memory Management**: With a Windows computer you are free to choose where you store your files, allowing you to keep your project secure throughout the entire development process, and avoid running into any unnecessary issues.
5. **Distributed Systems and Networks**: You could use a client-server distributing system here, since we'll have each client application rely on a single server application for our game, allowing each client application to be developed to the strengths of that client's system. We would also need a strong network would also be required, since the game depends on multiple clients connecting to a single server to play the same game.
6. **Security**: Windows 10 comes equipped with antivirus software, and users have access to a wide range of security programs designed for Windows. The computer automatically scans new files for malware, viruses, and security threats, and alerts the user when malicious software is detected.