

Draw it or Lose it

# **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 | 03/23/2024 | Pedro Martinez | Revision of Executive Summary, Design Constraints, Domain Model, Evaluation and Recommendations |
| 1.0 | 04/07/2024 | Pedro Martinez | Revision of Server Side, Client Side and Developmental Tools |
| 1.0 | 04/17/2024 | Pedro Martinez | Recommendations section |

**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 wants to expand their Android game, "Draw It or Lose It," to a more web-based platform. They are trying to make it more accessible on different platforms as well as trying to enable multiplayer. The solution involves a working and engaging web application that handles and supports multiple teams. It will also have a unique game with team names, to give it consistent gameplay. By having modern web technologies and following a very carefully structured plan. We can meet the client's requirements for a scalable, interactive gaming experience. The software design document will have an outline of the architecture, technologies, and development roadmap for the game.

## Requirements

*<* Please note: While this section is not being assessed, it will support your outline of the design constraints below. *In your summary, identify each of the client’s business and technical requirements in a clear and concise manner.>*

## [Design Constraints](#_2et92p0)

Cross platform compatibility: This is to let the game function on different platforms not just android.

Network Latency and Bandwidth: Since the game is dependent on the internet they need to really fous on the network usage to prevent lag.

Concurrency and Scalability: This is to help support the need of the multiplayer functions since they want a lot of players in there game at once.

Unique Identification: Since its an online game eac player/user needs to have a unquie name so the game cant tell the users apart.

State Management: To keep the game servers and clients on the same page meaning in sync they should be updates to maintain it.

Security and Privacy: To make sure the users data is safe and private from prying eyes and that there security isn’t at risk.

## [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)

Program Driver Class: is the entry point of the application with a main method. Its also uses ‘Gameservice’ to start the game.

GameService Class: is a service provider for game related operations and its also written in Singleton, which is demonstrated by getInstance(), to show and ensure that only one instance of gameservices exists. It also manages games, keeps track of the next game, player IDs, and provides ways to add and grab games

Game Class: is a individual game. It has a list of teams participating in the game and methods to manipulate game data, such as adding teams

Team Class: Its to show the team within a game. It contains a list of players and ways to add players to the team.

Player Class: Shows individual player and contains player-specific information.

Entity Class: Acts as a base class for game, team and player classes, by providing attributes like id and names. This shows the principle of inheritance, which helps the code be more organized as well as reuse some code.

SingletonTester Class: Its used to test the Singleton pattern and implementation in gameservices. Its to double down that only one Gameservices is made.

OOP: is being demonstrated by Encapsulation on the data and behavior. Inheritance is based on the entity class. Abstraction is the UML diagram. Lastly it’s the Singleton pattern, to ensure only one gameservice is the only one.

**"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** | Hosting a web application on a Mac offers benefits on a stable, UNIX-based operating system. They also have superior hardware and security, making it appealing for development with Apple's. But the higher cost, limited hardware as well as software and compatibility issues are still there. Macs still gives out a small to medium-scale deployments and performance, reliability, and a user-friendly prioritized, but scalability and cost-efficiency needs to be talked about | Linux is a highly favored for web-based applications due to its opens nature, which gives offering such as significant flexibility and customizability. It excels in stability, security. But, it has a steep learning curve for beginners and potential hardware compatibility issues, it has more community support then official | Windows Server offers a user-friendly interface and easy integration, its a strong choice for organizations invested in Microsoft. It provides support for .NET and other Windows product. It tends to have higher licensing costs compared to Linux and require more resources for performance | Hosting web-based applications on mobile devices. These devices can serve as cost-effective, easily carried servers for lightweight applications, especially in decentralized scenarios. But, they have limitations in processing power, storage capacity, and operation capabilities, which will be making them less suitable for high-demand applications and raising concerns for performance |
| **Client Side** | Developing software for Mac clients requires consideration of there platform's, such as the cost and time supporting Mac clients can be higher due to the need for purchasing Mac hardware for development and testing. But, the Mac platform offers a stable environment with a dedicated user base, making the investment worth it for applications | Supporting Linux clients in software development creates an understanding of the Linux pool of ecosystem. Which can significantly impact testing and compatibility efforts. The open source reduces software and licensing costs but may increase the time and expertise cost. But, the flexibility and configurability of Linux offer a strong foundation for creating highly customizable and efficient applications | Developing software for Windows clients involves navigating windows with widespread market penetration. The availability of development tools like Visual Studio, along with support for a wide range of programming languages and frameworks, can reduce the cost and complexity of development, although licensing fees for software and development tools may impact overall costs. The investment in time and expertise to develop for Windows can offer great returns. | Developing for mobile devices typically requires expertise in platform-specific languages and frameworks. The diverse mobile devices ecosystem uses testing and optimization, extending development time and a broader range of testing hardware. But, the many and still growing user base of mobile devices offers market opportunities, making the investment in development highly worth it for applications |
| **Development Tools** | For developing software on Mac, developers primarily use Swift and Objective-C as programming languages, with Xcode serving as the central IDE that provides comprehensive tools for building, testing, and deploying applications. Other tools and technologies frequently use Visual Studio Code. The use of these tools and languages, combined with Apple's and developer resources, gives the development of applications tailored for Mac | In the Linux environment, a wide range of programming languages is use and the open source community's preferences. Developers often use powerful and flexible IDEs along with a range of command line tools and package managers. The development tools and languages supports the creation of customizable and efficient software for Linux | For software development targeting Windows, C#, .NET Framework, and Visual Basic are commonly used languages, with Microsoft Visual Studio. It offers comprehensive tools for coding, debugging, and deployment. Other essential tools include PowerShell for scripting. | Mobile device software development uses Swift and Objective-C for iOS applications, with Xcode as the primary IDE. Cross platform development is used by frameworks such as Flutter. These languages and tools enable developers to efficiently create, test, and deploy mobile applications, catering to the many of the mobile user base. |

## 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 recommend that the best operating platform that works really well would be Unity.
2. **Operating Systems Architectures**: It would support Mobile Oss which would be android and ios. It would support Destop OS of windows x86 and x64 and the Web can comply games very easily.
3. **Storage Management**: Ussing google cloud storage would be ideal to use for Unity. Since they are capable on handing large data and user traffic. Which is important for multiplayer.
4. **Memory Management**: We would be using automatic memory management since that’s what unity uses. Which is collects all the garbage for the memory lifecycle to free up more memory. It also prevents memory leaks as well.
5. **Distributed Systems and Networks**: Unity network capabilities will manage the communications from different user in multiplayer. It will use real time data synchronization. It is also good at handling network connectivity as well as data persistence. Lastly they use the TCP or IP protocols for gameplay integrity.
6. **Security**: we will be using SSL encryption to protect data. Then we will also use a data encryption at rest with cloud storage. Unity uses these factors and even provides them to protect the users from a data breach.