

<Draw It or Lose It>

# **CS 230 Project Software Design Template**

Version 3.0

## 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 |
| --- | --- | --- | --- |
| 3.0 | <12/14/2024> | <Jian Wang> | <Analyze the characteristics of and techniques specific to various systems architectures> |

**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 wanted us to design a web-based multi-platform version of their existing Android-only game Draw It or Lose It. The game is based on the 1980s TV game Win, Lose or Draw. Our goal was to help the client list software design problems, design constraints, and propose solutions. And we will explain how to apply customer requirements to existing architectures for different operating platforms.>

## Requirements

*<* 1. Create a web-based application and to scale up to thousands of players.

2. Evaluate various platforms for their characteristics, advantages, and weaknesses for hosting a web-based software application and provide potential licensing costs to the client.

3. application must support players on iOS and Android mobile platforms, as well as traditional desktop-based operating systems. The application must be delivered as a modern, responsive HTML interface.*>*

## [Design Constraints](#_2et92p0)

<1. The client needs a server-style configuration for hosting the website and allowing it to scale up to thousands of players. It means the application needs to be more responsive and well-designed.

2. Compatibility issues, the team needs to learn about development tools for different platforms. May increase labor and time costs.

3. Third-party services may be used, and license fees will be incurred.>

## [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)

<Creating a game is done through the Game Service class. The Game class is introduced through the “add game” method, and the uniqueness of the game is achieved through get game id and name. Ensure that there is only one game in memory at the same time. In the Game class, the Team class is introduced through “add team” method, and then the player class is introduced through “add player” method. Similarly, the uniqueness of the team and player is achieved through id and name.

The entity class is the parent class of the game, team, and player classes. It defines the properties of other classes.>

**"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** | < The Mac system is architected in four layers: The User Experience Layer, the Application Framework Layer, the Core Framework Layer, and Darwin. this layered design makes the Mac system efficient, stable, and easy to maintain.  However, the mac system has more restrictions on third-party software, compatibility is not good enough, and the learning cost is high.> | < Modularized architecture, each user can use the system resources independently. Multiple tasks can be performed simultaneously without interfering with each other. However, Linux basically relies on commands to perform operations, compared to the graphical interface of windows, it is more difficult to operate.> | <The distributed system of windows is perfect for multiplayer online games. and windows is easy to use, low learning cost.  The disadvantage is that the server stability is not very good, but it is enough for <Draw It or Lose It>.> | <Easy to carry and ready to operate, but hardware is not sufficient to support hosted servers.> |
| **Client Side** | <Although the Mac operating system focuses on the user experience, but the hardware is expensive and the number of customers is small, and very few people use macs to play games.> | <Linux is an open-source operating system, which means you can modify, customize, and distribute the software freely.  But Linux has very little support for gaming, and at the same time, almost no home users use Linux  .> | <There are a lot of users using it, which means that many problems can be found to be solved. It is easy to use and the hardware is not expensive, which is why almost all home users are using windows.> | <The customer numbers are growing rapidly with the increased functionality of mobile devices, and the development environments for IOS and Android are very mature.> |
| **Development Tools** | < Xcode is a powerful IDE for developing Mac and iOS apps.Visual Studio Code, Atom, Sublime Text are highly customizable and extensible for writing all types of code.Eclipse is an IDE for developing in Java, C++ and other languages.> | < Linux programming and development tools are:Eclipse, LightTable, SublimeText3, Brackets, Make, Dart Editor, Vim, GCC. > | < C# (in combination with Visual Studio), C++ (in combination with Visual Studio or GCC), Python (in combination with PyCharm or Visual Studio Code), and Java (in combination with IntelliJ IDEA or Eclipse). Among them, the combination of C# and Visual Studio is especially good on Windows platforms.> | < AIDE (Android IDE), Dcoder, Pydroid 3, Juno, Kodex, QuickEdit, iSH Shell.> |

## 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**: < Recommended to use the windows platform, although you must pay some third-party fees, but its software compatibility is good, consider designing a cross-platform software, compatibility is the first consideration. Next is the cost of labor and time, most of the developers can easily use windows software, which can save a lot of unnecessary cost. Although Linux and Mac systems are stable and the third-party fees may be lower, developers are not very familiar with them. During the development process, additional Linux or Mac teams may be required, which increases a lot of costs.

For compatibility issues during development, choose to use cross-platform development frameworks, such as React Native, Flutter, Electron, which can share code and logic on different platforms, reducing development costs. Test the performance of the application in various mainstream browsers, including Chrome, Firefox, Safari, Edge. Comprehensive testing and debugging are key steps to ensure multi-platform compatibility. Listen to user feedback and opinions to understand their experience on different platforms, to improve and optimize the application in a targeted manner.>

1. **Operating Systems Architectures**: < Kerne, the kernel of the Windows system is the core part of the entire system, which provides various basic system services and resource management functions.

Subsystem, the subsystem of the Windows system is used to handle different types of applications. Each subsystem has its own PI interface to handle different types of applications.

Driver, the driver of the Windows system is used to control hardware devices, such as graphics card drivers, sound card drivers, network card drivers. Each driver has its own PI interface to control hardware devices.

Application, the application of the Windows system is used to handle various tasks, which includes multiple applications, such as Office applications, browser applications, game applications.

The DCOM distributed component of Windows enables processes on different computers to communicate with each other. When a client application needs to call a method on a remote server, it sends a request to the service port and then transmits the request to the server through the network. After receiving the request, the server executes the corresponding method and returns the result to the client. Through DCOM, data exchange and communication between different players can be realized, thus enabling the function of multiplayer games.>

1. **Storage Management**: < Windows Storage Management can manage storage space and optimize storage by releasing space on the hard drive. Disk cleanup function can be performed by deleting unnecessary files. In addition, the Windows Storage Management Console provides a graphical interface for managing computer storage resources.>
2. **Memory Management**: < Memory management keeps track of all memory locations. Specifies how much memory each process should be given. It decides which processes will be remembered and when. It tracks when memory is released or when it is shared and changes the status accordingly.>
3. **Distributed Systems and Networks**: < To enable communication between different platforms, games can be implemented by implementing centralized servers that can handle game synchronization, real-time updates, and message exchanges between players on different platforms. The server side has the right to make changes, while the client side only has the right to execute. This prevents the client from being modified, resulting in an imbalance in the game. The server side has the right to deny access when there is an error message on the client side, such as a duplicate name.>
4. **Security**: < Access to the server side can be controlled through user authentication and authorization mechanisms. In addition, data encryption technology can be used to protect customer information.>