

<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 | 03/19/2023 | Fisal Ikhmayes | Identified the requirements, constraints, and explained UML Diagram, and evaluated different operating systems. Also recommended the proper course of action to develop the Game. |

**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)

<Write a summary to introduce the software design problem and present a solution. Be sure to provide the client with any critical information they must know in order to proceed with the process you are proposing.>

My client The Gaming Room wants to develop a web-based game that serves multiple platforms based on their current Draw It or Lose It Game. The problem is this game is only available on the Android Platform. The Client does not know how to setup the environment, so my team will study the different platforms and provide the right option based on my client's needs and then code the application.

## 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.>*

**The business requirements** for the game are that the web-based game be available to multiple platforms which will help the client expand their user base and increase revenue.

**The technical requirements** are that

1. the game will be able to have one or more teams involved, each team will have multiple players assigned to it,
2. the game and team names must be unique to check whether a name is in use when choosing a team name,
3. Only one instance of the game can exist in memory at any given time.
4. The game should have a four-round play, with each round lasting one minute. Drawings are drawn at a steady rate and are fully complete at the 30-second mark, and if the team fails to guess the puzzle before time expires, the remaining teams have an opportunity to offer one guess each to solve the puzzle with a 15-second time limit.

## [Design Constraints](#_2et92p0)

<Identify the design constraints for developing the game application in a web-based distributed environment and explain the implications of the design constraints on application development.>

1. Cross Platform Compatibility: The game needs to be a web Based game with support for multiple platforms.
2. Team-Based Gameplay: The game needs to have support for one or more teams in each game. With each team having multiple players assigned to it.
3. Unique team and game names: The game must have a feature to check whether a name is already in use when choosing a team name, to avoid confusion among players.
4. Time Management: The game should have a four-round play, with each round lasting one minute. Drawings are drawn at a steady rate and are fully complete at the 30-second mark, and if the team fails to guess the puzzle before time expires, the remaining teams have an opportunity to offer one guess each to solve the puzzle with a 15-second time limit.

## [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)

<Describe the UML class diagram provided below. Explain how the classes relate to each other. Identify any object-oriented programming principles that are demonstrated in the diagram and how they are used to fulfill the software requirements efficiently.>

In this UML Diagram there is base class “Entity” which holds basic attributes and behaviors. Three subclasses “Game”, “Team”, and “Player” inherited from the Entity Class. There is a GameService class that is not inherited from the Entity class. There is a zero to many relationships between GameService and Game. There is a Zero to many relationships between Game and Team. There is a Zero to many relationships between Team and Player. Encapsulation is used in Entity, GameService, Game, and Team to protect the data using Private. There is a ProgramDriver class that is main driver. The program driver uses the SingletonTestor to test that there is only one instance of the game being created. Additionally, all these classes are in a com.gamingroom package.

**"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** | Because Mac servers are user-friendly, web developers can easily set them up and setup their web-based software.  The Mac operating system is not the best for hosting web-based apps. They lack the same degree of scalability as other systems and are not intended to be used as a server for hosting. Mac does, however, have a superior UI and a cleaner appearance. Developers can run web applications on a Mac server thanks to macOS's integrated Apache web server. The macOS operating system is private software and incurs licensing fees in order to use it. The expense of the license will vary depending on the macOS version and the kind of license necessary for The Gaming Room's particular requirements. Additionally, there may be licensing costs associated with any third-party web server applications used for hosting the web-based software application. Additionally, if a third-party web server application is used to host the web-based software program, licensing fees may be involved. | Linus is a fantastic option for storing software. It is very effective and scaleable. Additionally, it offers a stable platform for operating crucial applications and is secure. Compared to other operating systems, Linux can greatly lower deployment costs because it is free to use.  Weaknesses: Because Linux may not be as user-friendly as other operating systems, setting up and configuring the web server may be more difficult for coders who are unfamiliar with Linux.  With the help of web server programs like Apache, Nginx, and others, Linux provides a number of server-based distribution options. Developers can host and administer their web-based software applications by installing these programs on Linux servers.  Linux is an open-source operating system that can be used without a permission. However, third-party web server apps or other software used for the web-based software application may come with licensing fees. | Windows provides software apps with a user-friendly UI and a reliable, stable platform. Numerous computer languages are supported by Windows. Windows' security flaws are one of its weaknesses. It places a high priority on security and offers tools like Windows Defender and BitLocker for protection.  Microsoft technologies like the.NET Framework and Microsoft SQL Server, which are frequently used for web application creation, are also supported robustly by Windows.  With licensing fees for the server operating system as well as any required server software and applications, Windows can be more expensive than other operating systems.  It might need more updates and upkeep than other operating systems.  Depending on the particular Windows version being used and the licensing agreement selected, The Gaming Room, the client, may incur licensing fees for the server operating system. Although Windows Server licenses are usually purchased once, ongoing support and maintenance may incur additional expenses. | A software program works well on mobile devices because of their portability and accessibility. They do, however, have constrained storage, processing, and screen capacity. This may have an impact on output. Additionally, if the gadget is stolen, security may suffer. Mobile devices might not always have internet access, which can make it difficult to access web apps in specific places or circumstances.  The amount of data that can be saved on mobile devices for offline use may be constrained by their storage capabilities.  Typically, server-based deployment methods, where the website will be stored, are not provided by mobile operating systems. Web-based programs are usually stored on servers and accessed using a mobile web browser or a program designed specifically for mobile devices. There are typically no licensing costs for mobile operating systems for web-based applications, although app store fees may apply for distributing mobile-specific applications. |
| **Client Side** | Due to the requirement for specialized knowledge and experience, the cost of development may be higher when using a mac. Additionally, it might take longer to create and test for various clients. To guarantee compatibility with all mobile platforms and web browser platforms, developers must possess specialized knowledge and skill in the creation process. The programming languages used in Mac-based development, such as Objective-C and Swift, should be known to them. Additionally, developers must fully test the application on a variety of browsers and hardware. | Linux support for numerous software packages can be expensive and time-consuming. Additional resources may be needed for testing and integration in order to integrate with third-party suppliers seamlessly. The accessibility of materials and tools for software development is another factor. Linux has a large selection of development tools and libraries, but some of them might need special knowledge or training to use successfully. Creating applications for Linux platforms might come at an additional expense. License fees for particular technologies or programming tools may fall under this category. | Application development and testing for various Windows clients can be expensive and time-consuming, particularly when working with legacy systems.  It might also be necessary to have knowledge of different Windows platforms' applications. To make sure the program works with different mobile platforms and web browsers, extensive testing is necessary. To validate the program, developers should conduct unit testing, integration testing, and system testing. The layout and design of the application can be made to adjust to various screen sizes and resolutions by using responsive design techniques like CSS media queries, allowing compatibility with a variety of mobile devices. | Because every mobile device has different requirements and features, each one requires a substantial financial and time commitment. Continual upkeep and upgrades across numerous platforms can also be resource-intensive. It takes knowledge of several different programming languages, frameworks, and platforms to create a web-based software application that is compatible with mobile devices and all online browser platforms. To make sure the application is compatible with all platforms, it is crucial to try it across a variety of hardware and browsers. A variety of mobile gadgets and web browsers are needed for this. |
| **Development Tools** | A variety of tools and coding languages, such as Objective-C, Swift, and C++, can be used to develop software for Mac. Popular IDEs for Mac programming include Xcode, AppCode, and Eclipse with the CDT plugin. A team with experience in macOS development and proficiency in the necessary programming languages and tools is needed to create applications for the Mac platform. Since Xcode is a free tool offered by Apple, there are no costs involved with using it despite the fact that the technical requirements may be complex. To improve their creation process, developers might need to buy third-party tools or plugins. | The important programming languages for making Linux-compatible applications include C, C++, Java, Python, and Ruby. Several well-known IDEs for Linux programming include Eclipse, NetBeans, and IntelliJ IDEA. Expertise in particular computer languages, databases, web servers, and tools may be required of team members. Additionally, they might need to collaborate while using a revision control system, Multiple development teams may be required to manage various aspects of the development process, depending on the size and complexity of the project. It's possible that using some of the tools on the above list will cost money for authorization. For instance, some build tools and IDEs might have paid versions with more functionality or support. | Windows applications can be created with the help of the robust programming tool known as Visual Studio IDE. It has functions like testing, troubleshooting, and code editing tools.  These technological specifications may significantly affect a development team, particularly if they lack Windows application development expertise. The C# or Visual Basic.NET computer languages, the.NET framework, and the Windows API must all be mastered by the team. Several development teams may be required to handle various aspects of the project, including front-end development, back-end development, and database management, depending on the complexity of the software being created. The development tools needed to create Windows-based apps might come with licensing fees. Visual Studio IDE is a licensed product and requires a subscription to access some of its advanced features. | The cross-platform app development tool Xamarin enables programmers to build apps in C# and.NET that can be installed on Android and iOS devices.  The primary integrated development environments (IDEs) for Android and iOS development are called Android Studio and Xcode, respectively. These IDEs give users all the resources they need to build, test, and launch mobile applications.  The programming languages and tools that can be used to create applications for Windows include C++, C#, and Visual Basic. Some well-known Integrated programming Environments (IDEs) for Windows programming include Microsoft Studio and CodeLite. For the development of apps for various platforms, separate teams might be necessary. For creating Android apps, a team with Java knowledge would be needed, whereas for creating iOS apps, a team with Swift expertise would be needed. Additionally, there might be licensing fees for the programming tools. For instance, Xamarin's commercial version, which offers extra features and support, needs a license. However, using Xcode and Android Studio is free.’ |

## 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 advise adopting a Linux operating system as the server platform for extending Draw It or Lose It to different computing environments based on The Gaming Room's requirements. Popular open-source operating system Linux has excellent levels of reliability, security, and adaptability. Additionally, it is quite adaptable, enabling programmers to modify the system to satisfy certain needs. Linux is a great option for cross-platform development because it is also compatible with a wide range of hardware architectures and supports a number of programming languages.
2. **Operating Systems Architectures**: Linux offers developers a modular design that lets them pick the parts they need to build their system from a variety of options. The operating system's core, the kernel, and the user space, which houses the programs and services that run on top of the kernel, form the foundation of the architecture. Developers can interact with the hardware and access system resources like memory, storage, and network interfaces via a set of APIs that are made available by the Linux kernel. Each layer of the user space, including the shell, libraries, and application frameworks, offers a specific set of tools and services for creating and using applications.
3. **Storage Management**: I advise utilizing a distributed file system like GlusterFS or Ceph for managing storage. These file systems offer high availability and scalability by enabling data to be saved across numerous servers. They also support functions that can enhance performance and lower costs, such as data replication, data striping, and data tiering. Additionally, they offer a uniform namespace that enables data access from any cluster server, simplifying management and scalability of the storage infrastructure.
4. **Memory Management**: In order to optimize memory utilization and boost system performance, Linux makes use of a number of memory management strategies. These methods include memory mapping, which enables applications to access files as if they were a part of their memory space, and virtual memory, which enables the operating system to use more memory than is really available by paging memory in and out of disk. In order to prevent system crashes, Linux also employs a procedure known as the OOM (Out of Memory) Killer, which automatically kills off processes that utilize too much memory.
5. **Distributed Systems and Networks**: I advise implementing a distributed software architecture based on microservices to enable communication between many platforms. The application can be divided into small, independent services that can be deployed and scaled separately thanks to microservices. Since each service has an API that allows communication with other services, integrating it with other platforms is simple. I advise utilizing a service mesh like Istio, which offers features like load balancing, service discovery, and circuit breaking, to ensure dependability and fault tolerance. Utilizing technologies like VPNs and SD-WAN (Software-Defined Wide Area Networks), the network that connects the devices should be built with security and performance in mind.
6. **Security**: Any application must consider security, thus Draw It or Lose It must be created with security in mind. Access control, auditing, and SELinux (Security-Enhanced Linux), among other security capabilities offered by Linux, can be used to safeguard user data on and between diverse systems. I also advise adopting advanced encryption standards like TLS (Transport Layer Security) and AES to protect both data in transit and data at rest. To guarantee that only individuals with permission can access the system, authentication and authorization should also be used. Finally, to make sure the system remains secure over time, routine security audits and vulnerability assessments should be performed.