

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 | 08/18/24 | Christin James | Changes were made to the executive summary, design constraints, system architecture view, domain model, and recommendation of operating platform. |

**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 develop a web-based game working on multiple platforms based off their current game, Draw It or Lose It. This is currently working on Android only, so staff needs assistance in streamlining the development of the web-based version of the app. The game must have the option to have one or more teams involved and each team will have multiple players assigned to it. Game and team names must be unique, allowing users to see if a name is already in use when choosing team names. Therefore, the classes of the software; Game, Team, and Player, must have unique identifiers within them and there must be no duplicate values in each list of items.

## Requirements

The Gaming Room needs help in streamlining the development of the web-based version of the app as it is currently only working on Android. Cross-platform functionality is a must, so the necessary program adjustments must be made accordingly. The application must be able to generate images as the first step of the game, therefore the correct system of code must be implemented accordingly. There must be the option to create unique teams with multiple players assigned to them also. Lastly, the company needs assistance in determining the best operating platform (including mobile platforms) to deploy and run the updated application on.

## [Design Constraints](#_2et92p0)

To expand to web-based platforms, the process behind cross-platform compatibility must be taken into account in order for Draw It or Lose It to function on Mac, Windows, etc. The game must be programmed in a language which is compatible with common platforms that users will be playing on. As this is of top priority, the development team must also have knowledge of developing web-based applications. This ensures efficiency and will need to be a deciding factor in organizing teams.

With expanding to multiple platforms, there will be a change in infrastructure which the team must take into consideration to keep the same functionality from Android whilst making adjustments. Different libraries may need to be imported to enhance functionality for different systems. As there will be images that need to be generated, storage and resource use must be taken into consideration. The platform of choice must be able to accommodate the large file sizes of images, in addition to the program itself. To accommodate this, there must be storage checks within the program to determine if users’ devices have enough memory to download and operate Draw It or Lose It.

Inclusion of features within the program that enhance security must also be taken into consideration. With multiple platforms, users may be able to log in from different devices. There must be checks within the code to ensure only one platform can be signed in at any moment. This along with adjusting code to supplement existing security features of various devices will need to be done for well-rounded security. Libraries that accommodate authentication will need to be implemented to ensure secure logins. Along with this, authorization features to only allow admin to make changes to account information will be necessary. The appropriate libraries providing packages with these features will need to be determined.

The time it takes to make the above changes will also have to be considered, along with the requested budget to meet the needs of the game. As the platform to launch the updated game is to be determined, the decision to work on-site or cloud based will also need to be decided.

## [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 diagram illustrates the overall functionality of the various classes involved in the program. The classes Game, Team, and Player are subclasses of Entity. This is an example of inheritance in that the subclasses inherit the attributes and methods listed in the more generic Entity class. Each subclass has its own unique attributes and methods atop the inherited ones from Entity. Encapsulation is shown here in that subclasses contain methods of the parent class, though they’re not explicitly seen in the classes themselves, which also happens to fall under Abstraction. Polymorphism is also shown as the subclasses can use the methods listed in the parent class, Entity, to produce different outcomes with respect to the subclass. The subclasses also share direct relationships with adjacent subclasses along with multiplicity, shown by the line and “0..\*”, indicating zero to any number of object instances shared. This is important to note as it indicates that more than one team and one player can be created within the program. You can also note that the Team class’s addPlayer method uses a Player data type. A similar application goes for Game to Team, and GameService to Game. GameService also contains the singleton method, a crucial method in establishing only object from the class. A relationship between ProgramDriver and SingeltonTester is also indicated by the black arrow, indicating ProgramDriver uses SingletonTester. Here the SingletonTester class is used to determine that ProgramDriver is only running one object at a time. ProgramDriver is also where the main function resides. Lastly, GameService contains numerous accessor (getter) functions to retrieve private attributes listed in the class, which disables outside classes or users to access the private information. This is a necessary element for privacy and security measurements.

**"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)

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Mac has the advantage of fast processing speeds. However, there is a higher cost of licenses due to it being a proprietary OS. Costs also increase due to the requirements of higher-end hardware, although the OS is free with purchased computers. Performance is exceptional as Mac OS X is based on BSD system. It is easy to keep it running for days without needing to reboot, which is a plus for any project. Advanced security programs compared to competing OS’s make Mac more suitable for security measures.  MacOS offers server-based deployment methods. Of these are Mac OS X Server, a 64-bit computing platform that uses open industry standards and network security to deploy server-based applications. Another method is Apple Developer Enterprise Program, which is a set of tools for developing, testing, and distributing apps to users. To develop at the enterprise level there is a roughly $300 licensing cost to develop the app in the OS and distribute the app to Apple store. | Price is an advantage with Linux due to its free open-source OS. Linux also doesn’t require high-end hardware which also cuts costs. It is also reliable in runtime stability, as it’s based on the BSD system, making it preferrable for programming. Its processing power makes it exceptional for running a web-based app. The security systems are comparable to that of Mac, offering thorough protection from malware. Lastly, its Linux systems are more customizable, though not the most user-friendly.  Linux also offers server-based deployment methods. It is compatible with many hardware architectures, which is a plus point for teams that have a variety of these. It supports deployment to .NET servers. Its versatility also makes it ideal for server use in that you can deploy it on any environment.  Although it’s free to develop on Linux, there are licenses whose costs are comparable to that of Mac, ranging up to $450 at the enterprise level. Though it’s not necessary, these can offer protection for developers as updates and security features are included. | Price makes Windows somewhat more competitive than Mac, though it’s not as budget friendly as Linux due to it being a proprietary OS. Though improvements have been made, Windows has historically been known to have system crashes and reboots, making its hosting ability subpar to the other OS’s. It requires higher-end software than Linux, however users get quality support. It can be susceptible to viruses which need to be taken into consideration for web-based apps. A highlighted benefit is that Windows can run Microsoft web applications which cannot run on Mac or Linux servers. This is something that will be a deciding factor for clients that require these Microsoft technologies.  Windows offers a server-based deployment method called Windows Deployment Services which is a server role that can be deployed on computers running Windows server. The cost of licensing Windows Server depends on the edition and number of cores in the server’s hardware, ranging up to $200. | Mobile devices are generally more commonplace than the other systems, making apps more reachable to larger audiences. Hosting a web-based app is also low in cost and mobile web servers allow for hosting across numerous platforms which are both major advantages. The main disadvantage for these devices is the low support for any malware attacks, though fortunately these are very uncommon.  Mobile devices can also be used as a server. Although there is no set server-based deployment method due to the variety of devices available, developers can use servers compatible with their device. Licensing costs can vary according to the device, which will be up to the company to decide. |
| **Client Side** | Due to Mac’s proprietary system, expenses are a bit on the higher end. Advanced hardware also drives up costs. Time and expertise are dependent on the user’s familiarity with the OS. Those with increased experience with Mac will likely prefer it over other OS’s.  Cross-browser testing will need to be conducted during the development process. This ensures that the application works uniformly across multiple browsers and platforms. It’s best to test as soon as the first page of the application is ready to help identify compatibility issues early. OS testing will also be performed accordingly, with earlier testing ensuring efficiency in the development cycle. There are tools used for compatibility testing which check to ensure compatibility with various OS’s. | With Linux using open-source technology, the cost for it is very low. As it is not the most common OS, clients may need to familiarize themselves with its features. This may not be a feasible feat for most clients.  Various kinds of testing can be done to ensure that there is compatibility across platforms. These ensure that apps work uniformly with the various platforms that clients will be using the app on. Tools to test across platforms exist and should be used early in the development process. For example, Xcode’s iOS Simulator to simulate Apple devices, and Android Studio’s Emulator to emulate Android devices. Other tools also exist for more OS’s and browsers.  Enough time should be set aside just for this testing. | As Windows is also a proprietary system, its cost will be comparable to that of Mac. As with Mac, time and expertise will be dependent on the user’s familiarity with Windows. As its layout is of a specific setup, Mac users may need more time and knowledge to get integrated into the OS.  As with Mac and Linux, cross-browser testing will need to be done to ensure compatibility of the app across various browsers. This also applies for various platforms, for which tools exist to make the process easier for developers. Two examples are Xcode’s iOS simulator to simulate Apple devices and Android’s Studio Emulator to emulate Android devices. This process should be started early in the development lifecycle to ensure the app is moving uniformly across browsers and platforms.  Ample time should be set aside to test the aforementioned tools. | The advantage of mobile devices is the low cost. Due to their widespread use, there is a lower barrier to entry in the needed expertise to use the devices. However, time may be of issue as mobile devices are not as supportive in handling larger amounts of data as compared to the other OS’s. This can be a barrier to efficient work.  Cross browser-testing also applies to mobile devices. Incorporating this ensures that the developed app runs uniformly across browsers. The same applies for platforms. Tools are available which test for both, increasing the efficiency of the development process. These must be used starting from the beginning of development.  To ensure that the timeline of development runs smoothly, time should be blocked off to accommodate for testing different OS’s and browsers. |
| **Development Tools** | Swift is the most common language used in programming for this OS. Popular IDE’s include Visual Studio, Atom, and Sublime Text.  Visual Studio for Mac is an IDE that can be used to develop cross-platform apps for Windows, Linux, iOS, and Android. This also supports the Apple M1 chip and .NET 7, which can help with coding speed. Some features in it include IntelliSense, Refactoring, Version control, and Azure Functions. This well-rounded IDE will make it easy on the development team by only having to use one IDE, thereby cutting down time for development. Different teams may need to be formed to test each platform’s cross-compatibility; however, it may not be necessary due to the versatility of the IDE. Cost varies for Visual Studio for Mac, depending on the type of subscription, with the Professional subscription priced at $1200. | Many languages can be used in IDE’s that are commonly used in Linux, such as Visual Studio and notepad++. The advantage to using Linux is the ability to use various languages including both common purpose languages like Python, Java, etc., to CSS and HTML.  Visual Studio can also be used to develop cross-platform programs. Visual Studio (VS) Code can also be used, along with Spacemacs (that supports many languages), as possible IDEs to meet project needs. The variety of options gives flexibility for developers to choose the one that best fits their skills. Depending on the team’s skills, this can certainly expedite the process.  The plus is that VS Code is free for commercial use, which cuts down on costs. However, if the other IDEs are preferred, they can be bought. | A few languages commonly used to develop software for Windows are JavaScript and HTML. IDE’s include Visual Studio, Eclipse, and NetBeans.  Visual Studio (VS) Code can also be used to develop for other platforms. This along with Terminus, an open source Javascript program. VS Code is free of cost which is a plus point. Terminus can be bought at a subscription of $30 monthly (for business level). As there are a few options of IDEs to choose from, the development team has flexibility and can pick one that fits the skillset of the members. Having this flexibility can increase productivity by using an IDE that many are familiar with. | Java, Kotlin, Python, C++ and C# are some of the major programming languages used for developing Android mobile apps. A common IDE is Android Studio. Swift, C#, and HTML5 are some languages used for developing iOS mobile apps. The common IDE is Xcode.  IDEs such as React Native and Flutter can be used to develop products for multiple platforms. React Native uses a Javascript library with native mobile app development features. Flutter is a mobile UI framework from Google that allows developers to build apps for iOS and Android using a single codebase. Due to this feature, it may expedite the process for developers by only having to use a single codebase. It is also free, which is great for The Gaming Room, also making it quite popular for developers and product owners. As it has built-in functionality catering to developers, it can expedite the development process. |

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**:

The recommended operating platform for “Draw It or Lose It” is Mac. With reliability in performance of the OS, the team can be confident that there will be no system crashes or need for reboots for the duration of the project. As there was no mention of a strict budget from Game Room, the team won’t have to compromise on any features by using Mac. As the OS is based on the BSD system, its fast speed will enable the web-based app to be developed quickly and efficiently, leading to faster delivery of the product. Its top-tier security features will also prevent any malware issues for the team, which is an added plus to the overall function of the platform. Lastly, the option to use Visual Studio for Mac can help with the development of Draw It or Lose It to be used on other computing environments.

1. **Operating Systems Architectures**:

The core of the macOS contains the Darwin kernel, which is an open-source OS similar to Unix. This provides services like process management, security, file system support, and drivers (Kumari, 2024). XNU is another hybrid kernel used in macOS, which facilitates communication between software components and manages hardware resources (Kumari, 2024). There is also the user space that includes components such as system libraries, frameworks, and applications. The WindowServer process manages the graphical user interface (GUI). This process renders graphics, manages windows, and handles user input (Kumari, 2024). The Quartz Compositor is responsible for the compositing and rendering of graphics on the screen. This ensures smooth animations, window transitions, and more (Kumari, 2024). Core Graphics is the graphics rendering engine which supports vector graphics and imaging. Apple’s low-level graphics API is Metal, which allows developers to take advantage of the GPU for graphic tasks. Lastly, AppKit is the framework for building applications, and UIKit is used for iOS, both providing essential components for developing graphical UI, which are essential for Draw It or Lose It (Kumari, 2024).

1. **Storage Management**:

MacOS has a built-in storage manager which is its storage management system. This shows a rough breakdown of categories such as Documents, Apps, Systems, and more. However, large files that can be deleted by the users are not shown. Therefore, a proper third-party storage analyzer is a safe option to optimize storage in Mac. There are quite a few on the market, however they mostly share common features (one good option is CleanMyMac X) (Berveno, 2021). Many of such storage analyzers offer user-friendly interfaces that allow for easy management of storage. With these, quick scans of the Mac can be made to give a complete picture of users’ storage, with access to every file that’s sorted by largest folders. The option to remove files is provided, which is a simple and efficient way to manage disk space. Systems that offer features such as these are ideal to use with Mac.

1. **Memory Management**:

There are a few memory management techniques of macOS which will come into play when running Draw It or Lose It. A common feature of macOS’s memory management system is the high RAM usage. macOS features a fully integrated virtual memory system that’s always on, and this comprises the physical RAM itself and drive storage. The memory paging scheme increases performance by looking at apps’ memory usage patterns, then loading pages (memory blocks) which the apps might use. This pattern can be built over time with repeated use, as the system tries to optimize memory management. Developers can also specify for this which can come in handy for The Gaming Room. The system works in that it loads and keeps pages in memory. Pages that are more recently accessed are marked as active and when more memory is required by apps and there’s little memory remaining, inactive pages are removed. This maximizes system performance overall by reallocating memory to active apps. This type of memory allocation is of benefit to Draw It or Lose It, as the system will draw up the necessary memory needed by the app based on the patterns or developers’ instruction, and ensure that it will run as intended by prioritizing memory needed by the app.

1. **Distributed Systems and Networks**:

Communication between various platforms can be accomplished with distributed software, connected through a common network. The distributed apps can communicate through multiple servers or platforms on the same network from any location. The benefit of this is that information can be shared over the network from multiple devices, yet a crash of a single platform doesn’t affect the collective (Lutkevich, 2022). These distributed apps are broken down into either client software or server software with the former accessing the data from the server or cloud, and the latter processing the data. Cloud computing can be used instead of a server or hardware to process data. With this setup, if one app component goes down, it can failover to another component to continue running. Depending on the architecture, the exchange of data is determined. For example, in peer-to-peer, there’s no central server and each node in the network has the same responsibilities (Lutkevich, 2022). This is different to client-server architecture as the name implies. These various setups can determine how the game can operate, and the ideal architecture can be chosen based on preference. For Draw It or Lose It, the client-server architecture will be of most benefit as the program can be managed through the server. As the game deals with authorizing users during logins and assigning teams, the server end of the program is necessary to authenticate and authorize users. This will help to keep the game secure, and work to break down any security breaches from the clients’ end. Lastly, if there is any breakdown on the client end, the server is unaffected, and the game still runs.

1. **Security**:

There are quite a few ways to protect user information across various platforms. As logins will be used for users, an appropriate level of authentication would enhance security for users’ sensitive login information. Although platforms provide their own security features, it’s important to adjust code to enhance security of the program. This was implemented into the program for Draw It or Lose It using libraries and frameworks. The features of the security should also be explained to users in everyday language. Another way of protecting user information is through encryption (Visual Eyez Studio, 2017). In short summary, successful encryption changes the information the user inputs into a language which only the system can comprehend, thereby blocking hackers from accessing sensitive information. Transit encryption in the form of HTTPS can be used to secure usernames, passwords, and other important data transmitted from a device to the server. This especially comes in handy when the user is using an unsecured public Wi-Fi network, for example. In addition, staying up to date with security measures after the app is launched will ensure top protection for users. Updates should be offered to users along with user feedback requests to make necessary adjustments and fixes to vulnerabilities. Lastly, the built-in features of the platform must be understood and implemented. macOS encrypts data through FileVault, which prevents unauthorized accessing from outside parties (Larrea, 2024). Another feature is Gatekeeper, which asks users for confirmation before opening an application to prevent malicious software from being opened (Larrea, 2024). Two-factor authentication is also used, which can be implemented to prevent hackers from logging in on behalf of users playing Draw It or Lose It. These are some of the many security features offered in macOS, which in addition to program security measures, will ensure that users’ private information is protected for the web app.

**Citations**

Berveno, M. (2021, July 20). *How to manage storage on Mac*. Setapp. https://setapp.com/how-to/manage-storage-on-mac#:~:text=To%20open%20the%20manager%2C%20click,certain%20categories%20by%20clicking%20Manage.

Kumari, N. (2024, March 26). *Macintosh Operating Systems (Mac OS)*. Code 360 by coding ninjas. https://www.naukri.com/code360/library/introduction-to-macintosh-operating-systems

Larrea, D. (2024, March 29). *MacOS security features: A guide for apple it*. Apple MDM Device Management Software: Mac iPad iOS. https://www.kandji.io/blog/mac-built-in-security-features

Lutkevich, B. (2022, June 28). *What is a distributed application (distributed app)?*. IT Operations. https://www.techtarget.com/searchitoperations/definition/distributed-applications-distributed-apps

Visual Eyez Studio. (2017, November 8). *Developing a mobile app? follow these 12 tips for protecting and securing user data*. The University of La Verne Small Business Development Center (SBDC). https://lavernesbdc.org/news/developing-a-mobile-app-follow-these-12-tips-for-protecting-and-securing-user-data/