

<Draw-It or Lose-It>

# **CS 230 Project Software Design Template**

Version 1.1

## 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)4

[**Recommendations**](#_m8aleynsvzvc)8

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.2 | 12/11/2022 | Raymond Lozano | Updated Recommendation section.  Operating Platform and System Architecture suggestion updated.  Storage management storage suggestion updated.  Memory management updated with requirement on running the game optimally.  Security requirements updated for users who will be working on the server when the server is live. |

**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 has an Android app called Draw It or Lose It and wants to create a web-based version of their game. Draw It or Lose It is a team game where an image is pulled from a library and each team will try to guess the image that was rendered. Each game will last four rounds where the starting team will have 30 seconds to guess the image, and the remaining teams have 15 seconds if the team does not guess the image before the timer runs out.

## [Design Constraints](#_2et92p0)

* Each game has more than one teams.
* Each team has multiple players.
* Game and team names must be unique.
* Only one instance of the game can exist in memory at any given time.
* Game must be web-based across multiple platforms.

These are the current design constraints. The game is already on Android so it will need to be designed for iOS, Windows, MacOS, and Linux as well.

## [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 Player classes are all child classes of the Entity class, they all extend the Entity class to have access to the id, and name attributes. Each of the child classes will share the references name and id. The Game, Team, and Player classes all have multiplicity to them allowing there to be any amount of each. Each of the three child classes have an association relationship with each other allowing the use of their methods. The team has the players attribute which pulls from the Player class the list of players for the game, and the Game class has the teams attribute which pulls the list of teams from the Team class. Meanwhile the GameService class takes the appropriate list of games from the Game class to create and run the game.

**"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** | Popular among developers for web hosting.  Hardware and software built in-house allowing for more stable environment.  **Disadvantages:**  Hardware being built in-house means that it is not so easily replaced if any issues arise.  Apple does not provide server-based deployment, will need to use 3rd-party software. | Open source software, and considered the safest option out of all the operating systems.  Access to powerful hardware to go along with software.  Compatible with most hosting software.  **Disadvantages:** Not compatible with most windows applications.  Largest learning curve of any OS.  Due to being open-source, Linux does not provide server-based deployment, will need to use 3rd-party software. | Most dominating operating system.  Being the most dominating OS, it has access to the most software.  Access to powerful hardware to go along with software.  **Disadvantages:** Since it is the most dominating OS it is also more prone to viruses due to the amount of creators.  Windows servers require frequent rebooting, meaning downtime for users.  Very high cost for licensing a server.  Microsoft has three options for website hosting: Azure, Windows Server 2019, and Windows Server 2022. | Mobile server that you can take anywhere.  Hardware based on device. Not every mobile device is capable of being a server.  **Disadvantages:**  Potentially poor security depending on OS of mobile device.  Device overheating can be an issue.  Mobile devices generally do not provide server-based deployment, will need to use 3rd-party software. |
| **Client Side** | Due to being based on UNIX operating system, a bit more expertise is required compared to Windows.  Cost of device is higher than compared to other operating systems.  Development time may be somewhere in the middle compared to Linux and Windows.  To ensure cross platform compatibility, it would be a good idea to use a framework to handle the difficult work. Frameworks can take care of the difficult part allowing programmers to focus on other tasks. | Most expertise needed due to the near requirement of learning the in’s and out’s of knowing your hardware as well as use of the terminal.  Lowest cost out of any OS due to being open source.  To ensure cross platform compatibility, it would be a good idea to use a framework to handle the difficult work. Frameworks can take care of the difficult part allowing programmers to focus on other tasks. | Since Windows is a privately owned operating system, any web-hosting will require a license to use it.  As it is the most popular OS, there is little expertise needed, due to the amount of people with knowledge behind windows and the ease-of-use that Microsoft has made Windows.  With little expertise requirement, development time also goes down.  To ensure cross platform compatibility, it would be a good idea to use a framework to handle the difficult work. Frameworks can take care of the difficult part allowing programmers to focus on other tasks. | Cost is dependent on mobile device used as well as cost may directly relate to overall performance.  May take some time to setup and properly work.  Expertise may be minimal due to how user-friendly mobile devices are meant to be.  To ensure cross platform compatibility, it would be a good idea to use a framework to handle the difficult work. Frameworks can take care of the difficult part allowing programmers to focus on other tasks. |
| **Development Tools** | Access to Swift for development, which is specifically meant for Apple. Uses the C language. Multipurpose for apps and games.  Visual Studio Code is available as well for use as a code editor.  The impact of using 3rd-party software for web hosting, as well as using a framework is that it adds another layer of learning for the development team.  There are licensing costs for such software and frameworks, they usually come at a monthly price ranging anywhere from around $50 to $200+. | The most relevant programming languages for Linux would be C/C++, Java, Python, and JavaScript.  Linux has a wide variety of IDE’s and code editors to choose from.  The impact of using 3rd-party software for web hosting, as well as using a framework is that it adds another layer of learning for the development team.  There are licensing costs for such software and frameworks, they usually come at a monthly price ranging anywhere from around $50 to $200+. | Windows has a large variety of code editors and IDE’s to choose from.  Just about any language can be used within Windows.  With Azure or Windows Server, there may still be a learning curve even if it is developed by Microsoft.  Most likely the language will either be C++ or C#.  Pricing for Windows Server 2022 Standard is $1069. Standard 2019 around $800. These are one-time fees for the license. Azure is a monthly fee that can range to over $200. | Relevant languages for mobile development would be Swift/C and Kotlin/Java.  Swift/C is primarily used with iOS development making it the best tool for anything iOS related.  The impact of using 3rd-party software for web hosting, as well as using a framework is that it adds another layer of learning for the development team.  There are licensing costs for such software and frameworks, they usually come at a monthly price ranging anywhere from around $50 to $200+. |

## 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 Windows as the Operating Platform of choice to expand The Gaming Room’s game Draw It or Lose It. As noted above it is the most common operating system in the world meaning that most people already know and understand the system.
2. **Operating Systems Architectures**: Windows architecture is divided into three categories: Apps, System Services, and Operating System Kernel. When developing the app, it will communicate with the system services down to the kernel as it works its way through the hardware of the system. The Windows application programming interface (API) also affects applications and their ability to use the Graphical User Interface (GUI) as well as many other system services like accessing resources and audio.
3. **Storage Management**: Windows has access to Windows Storage Spaces, which increases storage space and helps protect against drive failures by grouping multiple drives together. As another form of backup to work alongside storage spaces, cloud storage can be used as well. Storage spaces will make virtual copies of the data so if one drive fails, or if a drive needs to be replaced, the data will not be lost.

As a form of physical storage in terms of reliability and speed using a solid-state drive (SSD) would be the best option. As SSD’s do not have moving parts within them like their hard drive counterparts, they are less prone to failure. Since SSD’s also use flash memories that means that data can be retrieved from them more quickly than hard drives as well. The amount of space needed for Draw It or Lose It would need to be the amount that covers the game itself, the data for the game (ex. pictures), and extra space for any updates that the game will need.

1. **Memory Management**: With Windows virtual address spaces, it allows Draw It or Lose It to store the pictures on a virtual address that the program will reference and call to when that specific picture is needed from memory. That way the game will be the only one to reference its assets from memory without other risk of other programs that are running at the same time to use reference them.

Utilizing cache memory will be an important aspect to ensure that the game will be as optimized as possible. While creating the game and running it on multiple devices, developers will need to optimize the way pictures are loaded for each player. By preloading the next picture into the cache, the CPU will be able to retrieve the set of data to load the picture immediately compared to loading it from the RAM or the storage.

1. **Distributed Systems and Networks**: Natively each operating system has communication with their same operating system for multiplayer games (ex. Game Sprockets on Mac, and DirectPlay on Windows). For these operating systems to communicate with each other, the networking side will need to built from scratch so that they can all share the same databases.
2. **Security**: Windows does come with security software such as Secured-core server, but it may be better to use third-party software instead or in conjunction with Windows security. This is mostly due to the number of holes that can be found within Windows. Adding another layer of security to Windows may increase the hardware load, but it may be considered worthwhile to use to ensure that Draw It or Lose It stays safe as well as keeping player information safe.

Windows firewall will be required to run a server using Windows. Developers will want to leave only the minimal required ports open to run the game. The fewer ports open, the least likely chance there are the be any intrusions. Draw It or Lose It will also need a back up in case anything were to go wrong. Authentication will also be required for users who work on the server.

One more important aspect of security is keeping Windows up to date. Windows periodically sends out security updates that are important for the health of the operating system, as well as patching any holes in its security.