

Win, Lose or Draw

# **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 |
| --- | --- | --- | --- |
| 2.0 | 06/1/2023 | Jessica Duft |  |

**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 company, The Game Room, would like to create a web based game, called “Draw it or lose it” that can be ran on multiple platforms. Currently, the game is only running on android. The premise of the game is as follows: The app will render an image from a database of stock piled image. The image will be a clue to the answer which could be a phrase, title or thing. Each game has four rounds of game time, each round lasting one minute each. The drawing will be rendered at a consistent rate and the team must attempt to solve the puzzle by guessing the correct phrase, title or thing. If the team fails to guess the puzzle before time runs out the other time has an attempt to solve for them. Whichever team has the most points at the end of the four rounds wins.

## Requirements

* A game will have the ability to have one or more teams involved.
* Each team will have multiple players assigned to it.
* Game and team names must be unique to allow users to check whether a name is in use when choosing a team name.
* Only one instance of the game can exist in memory at any given time. This can be accomplished by creating unique identifiers for each instance of a game, team, or player.

## [Design Constraints](#_2et92p0)

The game currently only runs on Android. The company would like for it to run across multiple systems such as Mac, Windows, and Linux. New code would need to be developed for each system or create code that would allow each system to inherit the already existing android code so that it can be ran on all systems.

## [Domain Model](#_8h2ehzxfam4o)

Entity acts as a superclass that accesses information from all other classes; Game, Team and Player. Game Service accesses elements from Game, Game accesses elements from Team, Team accesses elements from Player and Entity accesses variables id and name from all classes. The object-oriented programming principle used is called aggregation. Meaning that in an instance of a class, there is a reference to another class. This is all brought together by superclass Entity.

**"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** | Mac is a popular operating system known for it’s security and very few viruses. Mac offers easy to write commands that allows developers easy access to make changes such as reconfiguration. Mac can run Mac, Linux and Windows applications side by side. Mac offers frequent upgrades and high quality hardware options. However, Mac is more expensive than other options which could result in fewer users using it due to cost restrictions.  **The Mac OS Server does provide a server based deployment method that the game can be hosted on. However; this option is pricey, doesn’t have a lot of support, and is limited in third party apps and can only be ran on Apple hardware.** | Linux is also a very secured operating system where malware and viruses are caught early on. Linux is the most cost effective since it is free to install making it budget friendly. A disadvantage of Linux is that it could be difficult to find applications that would meet the client’s needs because Linux has the fewest application options of all other operating systems mentioned.  **The Linux OS Server does provide a server based deployment method that the game can be hosted on. Linux is cost effective, open source but has limited support and limited access to third party applications.** | Windows has the most software accessible for download. Windows operates on open and closed platforms. Windows is familiar to many and therefore often a preferred choice. Windows offers a wide range of choices and resources. However, Windows is not as secure as Linux or Mac which is a disadvantage but it is still relatively secure.  **The Windows OS Server does provide a server based deployment method that the game can be hosted on. Windows is a bit more pricey than Linux, but less expensive than apple. Windows is the most secure and has the most support and access to third party applications. It can be ran on any hardware.** | Mobile devices are portable and highly preferred for that reason alone. Because of the mobility, this option would allow for a larger amount of people to use the app. The disadvantages of the mobile devices are that they are not as secure as other options and may not be able to be used on ALL mobile devices.  **Mobile devices could be ran from either/or/both Apple and Android servers. Both servers provide a server based deployment method that the game can be hosted on. These servers are not as secure and are much more limited than the Windows server is. It is easy to use and learn but way less secure. It is more cost effective than the Windows Server but all in all an inferior option.** |
| **Client Side** | Moderate expertise and moderate time required. The cost would be comparable to the cost of developing in windows but higher than developing in Linux. However the cost of Mac OS hardware is higher than the cost of Windows hardware and should be considered.  **To ensure compatibility the development team must consider several things. They should consider creating and maintaining clean, neat, easy to read code so that it can easily be modified to fit platform requirements. The development team should use frameworks, such as Bootstrap, that make compatibility easier during the development process. The development team should test regularly across all platforms to ensure a bug free experience for users across all platforms.** | Maximum expertise and time required to develop using Linux. The cost however would be minimum unless you figure in development time then cost could go from minimum to moderate at least.  **To ensure compatibility the development team must consider several things. They should consider creating and maintaining clean, neat, easy to read code so that it can easily be modified to fit platform requirements. The development team should use frameworks, such as Bootstrap, that make compatibility easier during the development process. The development team should test regularly across all platforms to ensure a bug free experience for users across all platforms.** | Windows is easy to use so minimum expertise and time is required. Cost would be similar to the cost of developing in Mac but Windows hardware is more cost effective.  **To ensure compatibility the development team must consider several things. They should consider creating and maintaining clean, neat, easy to read code so that it can easily be modified to fit platform requirements. The development team should use frameworks, such as Bootstrap, that make compatibility easier during the development process. The development team should test regularly across all platforms to ensure a bug free experience for users across all platforms.** | Mobile devices are more portable making development more conveineint. However, maximum expertise would be expected as developing for mobile devices would require someone with extensive knowledge of app development since it's is different than web based development. Particularly developing the user interface would require someone with maximum expertise. However, this may be most cost effective since the game is already developed on android thus half way finished.  **To ensure compatibility the development team must consider several things. They should consider creating and maintaining clean, neat, easy to read code so that it can easily be modified to fit platform requirements. The development team should use frameworks, such as Bootstrap, that make compatibility easier during the development process. The development team should test regularly across all platforms to ensure a bug free experience for users across all platforms.** |
| **Development Tools** | **To develop using MacOS a developer could use swift on Xcode or similar. Could use a single development team. Licensing isn’t extremely expensive but compatible hardware is.** | **Linux can use visual studio and a host of languages such as HTML, CSS , and JavaScript for front end and Python, Java or Ruby for back end development. Could use a single development team. Licensing is cost friendly.** | **Would need a developer familiar with C# or C++ and could use Visual Studios. Could use a single development team. Licensing is somewhat expensive.** | **You would need a developer with knowledge of app development since it is different than web development. The game already runs on android devices so the work here is already half way complete; the company could have an app developer convert the code using Swift on Icode or another similar tool. Could potentially need multiple development teams. Licensing is cost effective.** |

## 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 using Windows as the next operating platform. Windows requires the least expertise as it is easy to use and has endless resources and IDES available. Combining a framework such as Bootstrap with the Windows operating platform would create an environment where the client could easily expand their game to several different platforms.
2. **Operating Systems Architectures**:

The windows operating system derives from the Windows kernel itself and other device drivers. System services interact directly with the system kernel, these systems are built in and do not need to be installed. The .NET framework is secure and widely known. This framework is used for the desktop itself, universal windows app and the windows app store. Using a client-server type architecture would work for this game since it is web-based and requires real time interaction. The client would send a command or input to the server, the server would run the command and send back the appropriate respond to the client. The company may consider using a cloud-based server such as Azure, which is compatible with the Windows operating system.

1. **Storage Management**:

Memory storage is another important factor to consider for our client as they develop their game, Draw It or Lose It?. Memory storage is where the data is permanently stored so that it can be transferred to users as they use the application. For this game specifically, we can think of where the file that contains all 200 images are stored permanently so that they can be transferred to any given user at any given time. Memory storage could be on a device, or it could be store on a cloud-based storage option. These files could be stored on some piece of hardware, such as the server or they could be stored on a virtual cloud-based option such as Amazon S3, Amazon EBS, Amazon Glacier, Google Cloud Storage, Cloud Filestore, Persistent Disk, Archive Storage, Blob Storage, or Azure Files. (Cloud Storage Cost Comparison: AWS vs. Azure vs. Google- Jay Chapel). All of these are compatible with Windows. Cloud based storage is a popular option in modern development since it is reliable, easy to access, and easy to purchase more memory as needed. Price is another factor to consider since essentially you would be constantly paying a monthly or annual fee for cloud-based storage but only paying once for hard drive storage or paying for it occasionally, when/if said device needs to be replaced or storage requirements outgrow the capacity of the storage available on the machine. Considering the large amount of data required by the client's applications I would highly consider a cloud-based storage option. The client would first need to calculate the total amount of memory needed, for the images, for the code itself and all other necessary files. After calculating total storage requirements, the company could purchase a cloud-based storage plan that fits their needs and requirements. If or when the company requires more storage, they could upgrade their cloud-based storage as needed. Doing so would prevent the company from purchasing a storage plan that is too small, causing user end issues, and from purchasing a storage plan that is too large, which would result in loss of profit as the client pays for a storage plan that is too large for their requirements. Windows 10 uses storage sense which allows you to manage files and storage on your hard drive and easily see where storage is being used. This allows the user to easily manage their storage. Windows also offers cloud-based storage called One Drive, which may be of particular interest to this company.

**Memory Management**: Memory management is an important concept when building an application and is especially important for the Draw It or Lose It game application being built by the client, since it requires the use of high definition rendered photos. Ensuring that memory is appropriately allocated means that these files can be transferred to the user in an efficient way that meets the client's expectations and requirements. There are 200 high-definition images that can be rendered to the user, each photo being about 8 megabytes in size. This means that for the image files alone the memory requirement is around 1600 megabytes. One thing the client could consider is compressing these images to a size that does not necessarily affect the quality of the photo. Compressing the photos to a smaller size would allow for a less demanding memory management requirement for the client. The client could also consider altering the code so that as little memory as possible is required throughout each phase of the game. The client could keep things how they are and integrate high-speed memory that can handle the large amounts of data being transferred to the user or they could make the suggested changes above so that their memory requirement wouldn't be as demanding, and they can still ensure that the user gets a top tier experience that loads quickly and effortlessly with easy to view and clear images. Using Windows, you could easily store these photos in any file and not be limited to the default photos folder. This allows you to keep your project together in one secure place on your computer. This would also allow the developer to easily open the files from the IDE while creating the game. Windows has built in memory management and a large development community that the development team could refer to if they had any concerns or questions about how to allocate memory most efficiently.

1. **Distributed Systems and Networks**: The company could develop on Windows and then use some cross-platform IDE, such as Visual Studios, to run the app on any device. Once the app is created on Windows the developer can upload the game file into the IOS, Linux, and Mac, etc. so that the game could be played on any device. The company would need to consider things like outages and in doing so be sure they have a strong server and a backup server or at least back up power to the main server so that there are not complete outages. If the company decides to use cloud-based server, they should consider a back-up cloud-based server so that if one server temporarily goes down, they can use the back-up one to prevent total outages.
2. **Security**: Windows has built in security protection software. The system will scan for malware, viruses, and other security threats in real time. As threats change the system updates to keep the system and user information secure. Providing a secure gaming environment is also extremely important for the client to consider. In order to prevent data breaches or other cyber attacks the company should consider using secure coding practices, data and password encryption, and multi factor user authentication.

**SOURCES:**

Chape, J. (2019, July 1). Cloud Storage Cost Comparison: AWS vs. Azure vs. Google. Jaychapel.Medium.com. Retrieved June 11, 2023, from <https://jaychapel.medium.com/cloud-storage-cost-comparison-aws-vs-azure-vs-google-844dfff3d324>

Sogbesan, T. (2022). The 5 Best IDEs for Programming on Windows 10. *MUO*. https://www.makeuseof.com/windows-10-best-ide/#:~:text=The%20Best%20IDEs%20for%20Windows%2010%20Users%201,IntelliJ%20IDEA%20. . .%205%205.%20Android%20Studio%20

TerryLanfear. (2023, February 7). *Secure development best practices on Microsoft Azure*. Microsoft Learn. https://learn.microsoft.com/en-us/azure/security/develop/secure-dev-overview