

<The Game Room>

# **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 | <02/18/2024> | <Matthew Kakareko> | <There were changes made to the executive summary, design constraints, domain model, evaluation, and recommendations.> |
| 1.1 | <02/25/2024 | <Matthew Kakareko> | Analyzed and recommended selection of software based on the characteristics of and techniques specific to various systems architecture. |

**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 project is a project where we are to develop a web-based game that is like “Draw It or Lose It”. Draw it or lose it is a game that is like the 1980’s television show Win, Lose or Draw. The game will be developed where the application will render images from a large library of stock drawings and these images will act as clues. The game consists of four rounds of play and each round lasts one minute. The images are rendered in slowly until 30 seconds where they will be shown completely. If the team does not guess the image; the other team will get 15 seconds to try to guess it. The project will be available on many platforms.

## [Design Constraints](#_2et92p0)

<Design Constraints:

* Security
* Storage
* Database
* Player Login
* Management for players, teams, and the game

>

## [Domain Model](#_8h2ehzxfam4o)

<In the domain model, entity establishes a relationship between the Game, Team, and Player classes. This means that they will inherit or get information from the entity class which acts as the superclass. There is a “has-a” (aggregation) relationship between GameService which has a Game and Game has a Team and team has a Player. This means that the instance of one class has a reference to an instance of another class. For example, Team has a reference to Players.

>

**"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 stable and secure which makes it a reliable choice. Its UI is limited in what you can do, but it is user friendly. Mac has poor compatibility with other software. It also contains flexible terminal commands. Apple servers can be purchased to run IOS for a cheap price. It is around $15-25, which makes it not a very expensive server to have for creating websites. This includes resources that help users create the server for MacOS. > | <Linux is good for web-based apps, it is low cost, and good for server-side applications. Linux is difficult to learn and use. Linux does have Command shell which is helpful for a simple server configuration as well as accessibility > | <Windows has tons of software compatibility, and it is easy to use. It is more expensive because of licenses, and it is prone to security risks. An example cost for Windows is that it costs around $250 a month to use and a lot more to own ($6,000). Windows also contains a command prompt. | <Mobile Devices are limited in what they can do because of their size, but they are good for web-based applications. One challenge is how mobile devices vary between users. For example, some might have iPhone or androids. Mobile Application servers can be expensive to use as they can be between $0 to $350 a month. A benefit of mobile is that almost everyone has a smart phone that can be used to play the game. > |
| **Client Side** | <Mac is known for quick development time. The downside to Mac is that the hardware can get expensive compared to other options. Developers need to be proficient in languages like C and Swift. Apple is not an open source, so the cost is low. > | <Linux is known for its diverse distributions and that it leads to longer development time. It is however, open sourced, which helps contribute to the low costs. Linux is not expensive, and it uses programming languages like C++ and Python. Not many people use Linux because not many people are familiar with it. > | <Windows clients are known for how easy they are to learn and use. It is also an expensive option. It uses languages like C# and .NET to develop such as Winforms. Windows is not open source. Many people use windows and are familiar with it, which means that it won’t take long to learn.> | <Mobile Devices are more difficult to implement than the others because of the various phone screen sizes and the mobile platforms. Developers need to know many languages for compatibility like Java and Swift. Most people use their smart phones, so it wouldn’t be hard to learn how to use it. > |
| **Development Tools** | <Mac’s developers use software such as Xcode. It is Apple’s IDE and it uses languages like Swift and C as well as UI software. Companies often use Git to track version control.> | <Linux has development tools like Visual Studios. It programs in C++ and Python. Linux has libraries that work to support front end/ UI. > | <Windows uses development tools like Visual Studio. It mainly uses programming languages that are C# and .NET to develop. Windows also has tools like Xamarin that give Windows the ability to develop cross-platform work. > | <Mobile Device development depends on the platform that the project is intended for. If it is Android then it will be developed in Java while if it is Apple, it might be developed in Xcode with the languages Swift and C. There are tools that enable cross-platform such as flutter.> |

## 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 most appropriate operating platform that will allow The Gaming Room to expand Draw It or Lose It to other computing environments would be Windows. Windows also has the largest user base as well as the most popular game development operating platform. This is because offers many software and tools that will help the developers develop the project as well as provide cross-platform options. We can use windows .net framework to develop the project as well as its UI elements. For example, developing in Visual Studios and using .net as well as C#. Windows is already integrated with other Android builds. We can use cross-platform options like React or Cordova with windows as a development environment. Windows also offers emulators such as PowerShell that could be used for testing on all the platforms. >
2. **Operating Systems Architectures**: <Windows developers commonly use software such as Visual Studios to develop the project. Windows separates the operating system into Kernal and User mode. Front end would be in the User mode while the backend, SQL Database, and hardware would be the Kernal. They can use languages such as C# for the backend and WinForms for the front end. Microsoft developed Windows to be able to offer many options to the user that range from file access to playing video games. Windows is well maintained and updated while also offers modern hardware and software for the user.>
3. **Storage Management**: <Windows storage can be stored on either hardware, in the cloud, or both. It is common for companies to store valuable information on their own hardware storage and add their non valuable information to the cloud where it is more susceptible to security risks. Having data on hardware runs the risk of losing data to things like power outages. Cloud based storage that is commonly used with Windows is Microsoft Azure. Azure offers data storage in many different regions which the user can choose. Having storage in multiple locations can be useful in case one region is experiencing power outage, for example if there is a natural disaster in that region, the other region can still be available. Cost options for Azure range from options such as data per second and time of storage. Where you can get charged by the amount of data you are storing for the time it is stored. This makes Microsoft Azure a great choice due to the pay for what you use concept which makes it a low-price option.>
4. **Memory Management**: <Windows would be a good option, because you can store information on the cloud and on the device itself. By having information that is supposed to be shared on the cloud, the users in the app will be able to view and store data that is meant to be shared. Some examples would be like the team names, player, etc.. It is a good place to store game results. The data stored on the hardware would allow for a place to store sensitive data such as passwords, user info, etc.. Windows also offers Windows 10 which is the newest version of windows. It offers great memory management without losing speed due to its efficient loading from memory process. It does this through the hardware by implementing disc paging. Windows will be reserving part of the hard disk and uses it as a extra RAM. This means that it splits the processes into smaller parts, and it only loads it into memory when it is needed. >
5. **Distributed Systems and Networks**: <Draw It or Lose It will be able to communicate between various platforms, this may be accomplished by using distributed systems and networks. The developers use tools and strategies such as load balancing to distribute the incoming network traffic while also having backup systems in place. Cross-platform compatibility relies on standardizing data and communications. By using Azure, we will ensure that the game is always able to run. Azure offers great uptime and by using multiple locations as our cloud storage we can ensure it is always running. It not only allows for constant uptime, but it can also store tons of data for a cheap price which will be needed for the more users of this app. By splitting the data to Azure as well as using hardware to store private or sensitive information the app’s data will be safe. >
6. **Security**: <Windows has a large user base which in turn means it is more prone to security risks. There are tools and 3rd party software that can help protect against viruses, but user caution will need to be taken as well. Such as user access restrictions. By using Azure, we have access to many security safety features. Azure offers Azure App Service which will greatly improve our security. It offers things like IP whitelisting, IP configuration, options for storage in a VPN. By using the IP whitelisting, we can ensure only those who should access the private data, should. Azure checks their password as well as requires an SSL connection. We can also offer 2FA for users to ensure that they are the ones using their account. It will act as a second security where they need to enter a pin that was emailed to their assigned email when they created the account or enter a pin that was sent to their phone number. >