

Draw it or Lose it

# **CS 230 Project Software Design Template**

Version 1.2

## 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 | 1/20/2023 | Matt Bandyk | First draft of Software Design |
| 1.1 | 2/5/2023 | Matt Bandyk | Updated Evaluation Section |
| 1.2 | 2/19/2023 | Matt Bandyk | Completed Recommendations |

## [Executive Summary](#_sbfa50wo7nsh)

Currently, Draw it or Lose it, a game by The Gaming Room, is available only on Android. The goal is to develop a web-based version of the game that could be executed on multiple platforms. The game will have one or more teams per game and each team can have multiple players on them. Only one instance of the game can exist in memory at a time, so each game, team and player will need to have a unique identifier, and players will be able to check if chosen game or team names are already in use.

## 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.>*

## [Design Constraints](#_2et92p0)

Design Constraints consist of:

* Developing a system to function on multiple platforms will require unique API’s to be implemented to ensure functionality and user experience.
* No duplicate game or team names can exist, program will have to validate inputs and allow for new names to be inputted if they already exist.
* To ensure only one instance of a game exists in memory, unique identifiers will be required for each game or team created.
* Due to web-based execution, server hosting must be identified and created
* Budget must be considered when looking into hosting services

## [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)

Looking at the UML class diagram below, we see there are a total of 7 classes within the gamingroom package. The ProgramDriver class holds our main method that will run the program. It utilizes the SingletonTester class to test that the singleton pattern is being utilized correctly in the other classes. There is also a base class called Entity which holds the common attributes and behaviors for the classes that inherit it. Those include Player, Team, and Game, all of which are child classes of the parent, Entity. The GameService class has a singleton process that insures only a single instance of a game can be running in memory at a time, and it has an association with Game which shows a multiplicity of zero to many, meaning there can be many games saved at a time. Game is then associated with Team showing the same multiplicity for number of teams that can be in a game. Team is associated with Player, also having the same multiplicity for number of Players that can be in a team.

**"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** | The Mac OS server hosting is a closed platform, meaning updates can only be provided by Apple. When new versions are released by Apple, you will need to pay to get it. The cost of hosting on Mac will carry a high price tag for licensing as well. Additionally, to host on Mac OS will require utilizing Mac’s, further raising the price tags as they are on the high end of cost for hardware. Mac would boast the most secure of the three. | Linux will provide the cheapest price option. Linux is free and open software and updates can be made on your own. Also, Linux would not require the higher end hardware that would be required to host on Windows or Macs. Licensing fee’s are non-existent, also adding to the reduced-price tag. | Windows, like the Mac, is a closed platform and would have the same constraints as the Mac. Price tag would be smaller than a Mac, but still more than Linux. Additional hardware requirements and a higher volume of updates being released by Windows could affect server status and licensing fee’s would fall in line with the Mac. Windows will also be the most insecure of the three. | While you can host the web-based software on mobile, it is not ideal. Mobile devices do not have the hardware or power to be affective hosting providers. Cost could be low, but scalability will be lost if you were to move forward with this option. |
| **Client Side** | When considering the client side on a Mac, Mac would run on the higher end of cost. There are many software development kits that could be utilized, but would have to be done on a Mac, also increasing the price. Development would be moderate in level when comparing Lunix and Windows, and same for level of expertise that would be required. | Once again, Linux will come in on the low end of cost, but not on the development side or time. Development on Linux will require the highest level of expertise to complete. | Windows would require the lowest level of expertise of the three when it came to development. Like the Mac OS, there are many SDK’s available, and cost to develop on Windows could be expected to be slightly less then on Mac. | Development on Mobile is possible, and it would provide a large amount of flexibility as developers can access and see updates remotely from anywhere. This would be on the high end of time and cost, and would be the hardest to implement out of all the others, Mac, Linux, and Windows. |
| **Development Tools** | Many of the same languages would be utilized across all the different platforms. For front end development, utilizing HTML/CSS/ JavaScript would be the best. For general development, utilizing Java, Python and PHP would be best. There are many tools or IDE’s that can be utilized to help in the development with these languages, like Eclipse, PyCharm and Visual Studio. Many of these tools have free licensing, but some IDE’s may require license’s to deploy commercially. If more than one language is utilized, specialized teams may be requried to complete the work, which could raise the cost of the software development. | Linux would be able to utilize all development languages and tools called out with Mac. It would also follow the same costing and personal requirements. | Windows would be able to utilize all development languages and tools called out with Mac. It would also follow the same costing and personal requirements. | Like the others, you would utilize the same development languages and tools to complete development for the mobile deployment as well. It would also follow the same costing and personal requirements. |

## 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**: My recommendation for The Gaming Room would be to utilize a FaaS (Function as a service) to complete the operating platform needs. FaaS is a cloud computing service, which means the physical hardware, virtual machine operating system, and web server software management are all handled by the cloud service provider. This will reduce the complexity that would be required to set up and maintain it completely by The Gaming Room. It is also a pay-by-use service, meaning you are only paying for what you need, rather than the expensive and time extensive process of doing it internally. There are many FaaS providers that could be utilized, but I would recommend Google Cloud as not only can it be utilized as your “Operating Platform” but can also be utilized for other areas of deployment, as discussed below.
2. **Operating Systems Architectures**: Utilizing Google Cloud means utilizing a serverless architecture. Google Cloud owns all aspects of the computing environment, including the hardware, middleware, operating system, and applications that would be utilized in the development process. Utilizing a front-end application like Google Chrome, you can access the FaaS to develop, maintain, and publish the application. An advantage of utilizing the Google Cloud is you are also utilizing Googles architectures in your work, meaning you are getting best in class support.
3. **Storage Management**: Google Cloud offers Google Databases, an efficient, low-cost storage option in which to store the many images that will be utilized within the application. It provides easy scalability and reliability to host and maintain The Game Rooms data. It also allows for easy management of the database, allowing for adding/removing of images in the future if the need ever arises.
4. **Memory Management**: By utilizing Google Cloud Functions, you have full control over the allocated memory associated with each function call. As this is a FaaS system, each “function” is its own individual memory usage. Utilizing the virtual serverless approach, each function request is completed and then dropped, meaning you will never run out of memory as it is operated through the google service. The other aspect of memory management is with the application itself. When developing the application, utilizing a virtual memory approach, which would allow the allocation of memory needs when needed as well as freeing memory when it is no longer needed. Virtual memory enables an operating system to compensate for physical memory shortages, temporarily transferring data from RAM to disk storage, which allows secondary memory to be treated as primary memory.
5. **Distributed Systems and Networks**: By utilizing the FaaS through Google Cloud, the application will be hosted on VM(Virtual Machine)/servers, which will allow clients/users the ability to access the game via an internet connection. This will also allow the various systems to communicate with the application to send requests and receive answers utilizing the HTML code that has been developed in the front-end development. Utilizing this system, google cloud will ensure that the application is always up and running, and if the client is connected to the internet, they can access the game from any system that the application has been developed or deployed to. Their host services are also distributed cloud-based, meaning that users are able to access from multiple locations based on their physical location, which can reduce the chances of user time-outs or server downtime.
6. **Security**: Security is always a major concern when developing an application. To ensure that it is secure and your client’s information is safe, there are many things that should be implemented. By utilizing a strong secure software development framework and utilizing best practices throughout the development cycle from the beginning, you can mitigate the risks that can arise. Leaning into authentication and authorization, and the principle of least privilege, you can ensure that users of the software are verified before access and ensure that they can only access what they are allowed to. This can be done with role-based access control by specifically indicating exactly what each user role is able to interact with.