

Draw It or Lose It

# **CS 230 Project Software Design (Gabriel Romero)**

Version 1.2

## 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 |
| --- | --- | --- | --- |
| 1.1 | 03/20/2022 | Gabriel Romero | Initial versioning of Draw it or lose it application for The Gaming Room |
| 1.2 | 04/03/2022 | Gabriel Romero | Evaluation |
| 1.3 | 04/13/2022 | Gabriel Romero | Recommendations |

**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)

Draw it or lose it aims to be a simple game in which a team or teams compete at guessing a what the drawing provided is (phrase, title, or thing) as it is rendered over 30 seconds. The game will consist of 4 rounds at 1 minutes each giving each team 30 seconds to guess their drawing. If they cannot the opposing team has a chance to steal the points with a 15 second guessing window. This option will not be available if its just one team. Each game will be able to have one or more teams, and each team will consist of one ore more players. Each game instance will be unique along with players, team names and game names.

## [Design Constraints](#_2et92p0)

* Web based game based off the current android only version.
* The image rendering needs to be finalized to ensure consistent experiences through many play throughs.
* All identifiers must be unique and can cause issues if not handled correctly.
* Every team will have from one to multiple players
* Must run on several platforms

## [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)

Given the current UML models we can break down the relationships between objects. To start we are given the Entity Object which each of the following inherits from: Player, Team, Game. These three objects will all have similar methods and variables provided by the parent Entity. Between them, A GameService can be associated to 0 to many Game objects, while a Game object can be associated with 0 to many teams, and teams can be associated to 0 to many player objects.

**"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 OS based off Unix systems so it would have a very similar setup to that of Linux.  Terminal is flexible and plenty of documentation on setups and proper maintenance best practices.  Still making its way into mainstream datacenters and networks. Does not have the track records of the other two main OS options when it comes to hosting. Looking at the cost of hosting for a dedicated hardware setup we will be looking at 80 to 300 dollar per month and about 5 to 15 a month for shared hosting. This pricing model is consistent throughout each of these entries, as we are not hosting it ourselves. | Very similar to Mac but price point would be greatly reduced. Many open-source projects can be leveraged to setup server side properly.  Very secured back-end environment and widely used. Great for intranet focused projects and access, can be cumbersome when trying to provide too multiplatform.  Looking at the cost of hosting for a dedicated hardware setup we will be looking at 80 to 300 dollar per month and about 5 to 15 a month for shared hosting. This pricing model is consistent throughout each of these entries, as we are not hosting it ourselves. | This OS has an extensive library of software available.  Widely used for web hosting and application hosting.  Ease of use due to the amount of information available at setup.  Unfortunately, heavily targeted by nefarious peoples, so security and virus protection would have to be double downed. Looking at the cost of hosting for a dedicated hardware setup we will be looking at 80 to 300 dollar per month and about 5 to 15 a month for shared hosting. This pricing model is consistent throughout each of these entries, as we are not hosting it ourselves.  . | Hardware issues would cause bottlenecks if primarily focused on using “mobile” equipment for hosting.  High portability enables ease of migrations if needed.  Could also end up being cost effective if the application that is being hosted is very simplistic in use and doesn’t require the heavy computing power of other applications. Looking at the cost of hosting for a dedicated hardware setup we will be looking at 80 to 300 dollar per month and about 5 to 15 a month for shared hosting. This pricing model is consistent throughout each of these entries, as we are not hosting it ourselves. |
| **Client Side** | The main takeaway would be compatibility with client-side browsers. Since this is a web-based app that would take priority.  Would not take much expertise on the client side to navigate as long as they are able to access the application via the web as previously stated. To ensure that we are compatible through most web browsers a few steps must be taken. Keep the code as simple and non-unique as possible. Do not use OS specific commands and focus on generalizing. A framework would help in these instances. And perform browser tests for each OS possible. | Not a very user-friendly experience. Would require extensive training on packages and installation best practices to keep the machine up to date as well as compatible with the application versioning. Would not cost as much for Mac or windows as there are even free options for client-side use but a large learning curve. To ensure that we are compatible through most web browsers a few steps must be taken. Keep the code as simple and non-unique as possible. Do not use OS specific commands and focus on generalizing. A framework would help in these instances. And perform browser tests for each OS possible. | Same as mac, with minimum expertise required. Would need to understand basic computing skills.  The focus would again lie in the realm of the browsers and navigators to the web application. To ensure that we are compatible through most web browsers a few steps must be taken. Keep the code as simple and non-unique as possible. Do not use OS specific commands and focus on generalizing. A framework would help in these instances. And perform browser tests for each OS possible. Repeated because it applies to all 4. | The expertise ceiling is a little higher here as there are several different types of mobile devices that need to be accounted for. There needs to be an understanding of what will be used even if compatible with browsers within mobile devices would that be the best option?  Application already exists for android so that would take care of half or more of the market share, but time would need to be spent focusing on all other devices. To ensure that we are compatible through most web browsers a few steps must be taken. Keep the code as simple and non-unique as possible. Do not use OS specific commands and focus on generalizing. A framework would help in these instances. And perform browser tests for each OS possible. |
| **Development Tools** | Since this is a web-based application the language is not as important as the main project will be hosted on the server side of things. Although there should be knowledge of HTML, CSS, and Java. Along with the ability to work with them in tandem.  If working from mac Eclipse works well on there for Java development and comes with some great packages that can be useful. | Like Mac but also noting that the application is web based. The same languages should be accounted for and understood. VS Code works well on distributions of Linux along with any forms of packaging required. The application could also be built using any language that is required. I do not see this being an issue with the languages but more of how-to setup the teams. If we want to ensure best setup, we will need to ensure all bases are covered with browsers on other OS systems. | VS Code, Eclipse will work just fine within windows. I do recommend using Eclipse for any Java focused projects as it is easier to build something deployable from the get. Tons of compatibility with languages and even has an option to setup a subsystem based in Linux, so you get the best of both worlds while still working in a windows environment. Same knowledge of HTML, CSS and Java would be recommended. | The two main languages would be swift and android suite. This would be more important if the application was being focused on as an open platform application. As it stands it already had and android version and there are plenty of ides and languages that can help convert over to iOS compatible code. Such as react native. What it will come down to eventually will be testing on all major platforms. |

## 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 focusing on a windows-based environment for this project as it would be the easiest to get into and setup in my opinion. There are tons of compatible IDEs that can get the job done and this software put together. It would also help compatibility of the client side as most computers are running windows OS.
2. **Operating Systems Architectures**: Windows server provides a large tool set for setup and maintenance for applications such as Draw it or Lose It. Having setup some windows servers myself, I can say that the process is intuitive. Over the years Microsoft have upped their game when it comes to their GUI for their server software. This will come in handy when setting up server maintenance, scheduling tasks and running updates.
3. **Storage Management**: Storage Management Provider or SMP for short would be the focus on the windows side. This allows customization on how the drives on the server will be setup along with any redundancies for backups. A RAID 5 setup would be most beneficial as it gives us data redundancy, speed and is only held back by the smallest drive in the array.
4. **Memory Management**: Since the focus would be on having this setup in a distributed system environment, we can have the images and answers stored on a separate server all together. This way the main server isn’t bogged down with multiple calls while running the game services. We can then have that dedicated server feeding the images and alleviate the need for intense memory management and allocation.
5. **Distributed Systems and Networks**: Given that there will be multiple devices communicating with the hosting server, redundancies will need to be in place in case of outages. Having our application split into a distributed systems environment allows the leveraging of network resources we would have had to pour into one physical location. This also means that compatibility for outside users would not really be an issue. As the distributed system acts holistically to provide a seamless service. On the user end it would seem as if it was one seamless app but, in the background, we are gathering resources from across our network. Although having the systems separate like this leaves us at the mercy of our network, which would need to be secured and maintained to the highest degree. This also means more awareness for malicious attacks aimed at our network.
6. **Security**: Over the years windows has made great changes to their security and group policy when it comes to windows server software. They can now be more granular when it comes to security. This means that network traffic can be more heavily monitored and secured. There will need to be a time set aside for system maintenance, such as updates and restarts of the servers. There are a bevy of options when it comes to antivirus and antimalware software which will double down the security when needed but it would also come at a computing cost when running scans. This would also be something that needs to be scheduled during low application use times.