

<Draw It or Lose It >

# **CS 230 Project Software Design Brayall**

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 | <09/16/23> | <Davis Brayall> | <Brief description of changes in this revision> |
| 1.1 | <10/13/23> | <Davis Brayall> | <Determine software considerations and evaluate platform characteristics.> |
| 1.2 | <10/18/23> | <Davis Brayall> | < |

**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)

Our client, The Gaming Room, would like us to develop a web-based version of their game, Draw It or Lose It. We need to develop the game so that it has multiple teams, multiple players per team, allow only unique team names, allow players to check whether a name is in use, and allow only one instance of the game to exist at any given time.

## Requirements

The Gaming Room, our client, needs the game to function the same across all platforms, be cross platform compatible, within budget, and all the software requirements mentioned within the summary.

## [Design Constraints](#_2et92p0)

* Web based language for use in any environment.
* Multiple teams and players with unique names.
* Multiple players per team.
* Players need to check if a name is available or already used, with a singleton pattern.
* Only one instance of the game at a time.

## [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 following UML diagram shows ProgramDriver and SingletonTester, Entity and Game, Entity and Team, Entity and Player, GameService and Game, Game and Team, and Team and Player. The connections between GameService, Game, Team, and player show there may be classes of each in the other. Other arrows show inheritance.

**"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 includes built-in Apache webserver, an open-source software that has been around since the mid-nineties. A suite of command line tools and commands to configure or access the server.  Cons: Runs on limited hardware configurations, and limited software options.  > | <Linux is a more cost efficient and stable option, compared to windows and mac systems. It has a strong command system and webserver as well as can run for long periods of time. > | < Windows is the platform that most users are familiar and comfortable with. Windows also has a plethora of web extensions and SQL support. There is also a lot of software for windows not found on other systems.> | <Mobile devices are not a viable option to host the server. They are limited on computing power, memory, as well as software redevelopment issues.> |
| **Client Side** | <Mac’s cost more up front than the other operating system options. There is a moderate learning curve for operating on these systems.> | < Linux is the most cost-efficient option to use. It has many open-source programs that work in unison with the system. One downside will be that it might take longer to learn the Linux system and get the software ready> | < Windows has the most support for web-app and website development compared to other operating systems. It also has less of a learning curve comparatively.> | <The biggest factor is the ease of access and use. Requires the development team to refactor the app for the size, screen capabilities, and hardware of many different mobile devices. The app needs to be very intuitive due to the size and interface of the devices.> |
| **Development Tools** | <VSCode Xcode IDE, Homebrew, and Unix are a few options that can be used on a mac. JavaScript, Jana, CSS, and HTML are some of the languages that can be used.> | <VSCode, Atom, Visual Studio, and Git are some of the tools we can use. The languages are about the same, for all the computer options.> | <VSCode, Git Bash, Git, and Node are a few options. Again, the languages for the three operating systems are almost identical these days.> | <There are several web browsers including Firefox, opera, and chrome. These will allow for access to recourses to programs and test. JavaScript is the most beneficial option for mobile devices.> |

## 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 recommended operating platform is Linux. It is open source, supports a variety of IDE’s, is used by many developers reducing learning curve to use it, and is flexible.>
2. **Operating Systems Architectures**: <Linux has the following components to its architecture, kernel, system library, hardware layer, system, and shell functions. These aspects help the user do the different tasks. The way Linux is designed allows for easy use, access, and readability.>
3. **Storage Management**: <I recommend using a cloud-based storage platform, with the game being web-based, ease of setup, ease of use, ease of upgrading, and so on.>
4. **Memory Management**: <The application will be operating within java, thus will be using its memory management systems more so than anything else. Linux is a versatile system, with many memory management features that can be used if needed.>
5. **Distributed Systems and Networks**: <The game will be java based and played through the web. Any user with any operating system and internet capabilities will be able to access and play the game.>
6. **Security**: <Most if not all cloud-based servers provide security for the servers. Our software will have its own security features. We will also have to do regular updates to ensure the security features are up to date with current security threats.>