

The Gaming Room: Win, Lose, or Draw

# **CS 230 Project Software Design Template**

Version 1.0

## 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)3

[**Recommendations**](#_m8aleynsvzvc)5

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 09/16/2021 | Hunter Richards | Created the Executive Summary, Design Constraints, and Domain Model |
| 1.0 | 10/03/2021 | Hunter Richards | Created Evaluation Portion |
| 1.0 | 10/12/2021 | Hunter Richards | Created Recommendations Portion |

**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 is seeking to create a web-based game that caters to a variety of platforms. The game has multiple rounds, each a minute long, and revolves around guessing rendered stock drawings correctly. If incorrectly guessed, the other team will have a 15 second timeframe to correctly guess the stock drawing. The game is based off of The Gaming Room’s android app.

## [Design Constraints](#_2et92p0)

* Adaptable screen resolution
  + The screen resolution would need to adjust for unique device screen sizes. Thus the program must account and adapt to these conditions.
* User interfacing
  + Different clients must be able to join the same game and interface with other players
* Dynamic programming language
  + The application must contain a language that allows for dynamic content on a webpage that runs in the browser (such as JavaScript)
* Web browser accessibility
  + The web application would need to properly interact with various web browsers and adjust accordingly (i.e. Safari, Google Chrome, Microsoft Edge, Firefox)

## [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 below diagram shows Entity() as a base/foundation class for other classes to inherit from. The Game(), Team(), and Player() class all inherit from the Entity() class. These three classes all bear associations as they rely on one another to be present for program stability. The GameService() class acts as a driver for the many concurrent games running. The ProgramDriver() class is the primary driver of the entire program; thus main() is located here. The SingletonTester() is merely a tester to ensure that the Singleton pattern is active within the GameService() class.

These classes utilize the four OOP principles. The Game(), Team(), and Player() classes inherit from the Entity() class. This directly correlates to **Inheritance** which is a key OOP principle. The program also uses **Encapsulation** through the use of multiple classes (such as Game(), Team(), Entity(), GameService()). This allows for data hiding and increased flexibility when it comes to modularizing the code. Furthermore, **Polymorphism** is present among the Entity() class. As seen in the UML diagram, the Team(), Game(), and Player() classes are all an ‘Entity.’ They override certain characteristics of the Entity class (specifically the toString() method) which is one of the forms of Polymorphism. Likewise, **Abstraction** is present among the Entity() class and its inheritors. The Entity() class is an abstract class that concatenates to concrete classes (such as Team(), Game(), and Player() classes). This aids in standardizing Team(), Game(), and Player() with each class having an accessible id and name.

**"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 rarely used to host web-based applications. The OS is restricted in its hardware/software compatibility. This operating system does offer a built-in web server with Apache and PHP. Yet, of course, you would be limited to the hardware capabilities of the device (Apple is very picky with its hardware requirements). Furthermore, Apache is open-source, thus no licensing is required to run a webserver. | Linux and its distros are extremely configurable and have excellent options regarding hosting web applications. Linux is also open-source and has no associated costs. The most widely used OS for hosting. Ubuntu, a linux distro, is an excellent option as it is widely used, open source, and free. Other great distros include CentOS, Debian, and Slackware. | Extremely versatile operating systems with platforms dedicated to server hosting (i.e. Windows Server). Excellent tools and management options with wide-ranging industry use. However, Windows Server does require a license ranging from hundreds of dollars to over five thousand USD for their enterprise version. | Not built for hosting web service based applications. Yet there are options for hosting. I-jetty is an open source software which is capable of running java-based web content. This application is scalable to allow for tens of thousands of HTTP connections with far more web socket connections. However, Jetty is not an operating system yet merely an application to host web applications on mobile devices (specifically Android). Jetty is open source and free to use at any scale. |
| **Client Side** | For the Draw It or Lose It application to run successfully on the default MacOS web browser, compatibility for Safari needs to be taken into account. The application would need to be thoroughly tested in Safari using a variety of screen sizes to emulate client devices. | For the Draw It or Lose It application to run successfully on the default Linux web browser, compatibility for Firefox needs to be taken into account. The application would need to be thoroughly tested in Firefox using a variety of screen sizes to emulate client devices. | For the Draw It or Lose It application to run successfully on the default Windows web browser, compatibility for Edge needs to be taken into account. The application would need to be thoroughly tested in Edge using a variety of screen sizes to emulate client devices. Internet explorer wouldn’t need to be accounted for as it is mostly deprecated and rarely used. It should be noted that Chrome has the highest market share of all web browsers used. Thus it is extremely important that the Chrome browser is tested as well. | For the Draw It or Lose It application to run successfully on mobile devices, compatibility for multiple mobile web browsers need to be taken into account. Different mobile devices use different default browser applications. Android devices may use Chrome as the default browser while iOS devices may use Safari. Screen size would especially need to be taken into account for each mobile browser as it could majorly impact the web application’s appearance due to such a wide range of screen sizes. |
| **Development Tools** | MacOS has their own integrated development environment called Xcode. Xcode does have an annual fee of 99 USD to 299 USD. Xcode tends to be a less common IDE which may cause slowdown in the development process (due to lack of documentation and average familiarity). Xcode supports Swift (Apple’s own language), C, C++, Java, Applescript, Python, and more. It should be noted there are third-party alternatives (such as Eclipse or Netbeans) which may be a better choice for MacOS. Eclipse is open-source and free to use for developers. | Linux has many different options available to it. Vim comes pre-installed on Linux and has excellent syntax highlighting. Many consider it an IDE due to its advanced and resourceful features (modal, customizable, keystroke combinations). Vim is free and open-source which makes it an excellent contender for development. It can also be fitted to support most languages (including Lua, Java, Python, C, C++, etc). Other third-party IDE’s are available to Linux which are also excellent choices (Eclipse, Netbeans, Atom, Sublime). Linux, in particular, has many wide-ranging/open source/free IDE’s available. | Microsoft primary IDE for Windows is Visual Studio. Visual studio is an excellent IDE. It has many inline integrations that make learning/debugging extremely efficient. The license for Visual Studio is a bit pricy at around 45 to 250 USD per month. Yet this license comes with the added benefit of Azure DevOps which is a powerful tool for team software development (similar to Github yet catered towards enterprise development). It should be noted, however, Visual Studio doesn’t typically support certain languages such as Java (There are workarounds). The current project would need to be remodeled into a C – based language for this IDE to be practically considered. As always, third-party IDE’s are always available to Windows (especially so). | Development on mobile device OS’s is a bit tricky as it is uncommon. However, Android Studio is an option. Android Studio supports C++, Java, and more with extensions. The tools and features pale in comparison to other IDE’s. It is open-source and free for developers which is definitely a plus. Multiple teams would be needed as there are many different mobile platforms available (i.e. Xcode for iOS 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**: I believe the best operating system for The Gaming Room’s Draw It or Lose It application is Ubuntu Linux. Ubuntu comes with state of the art security directly embedded within its OS. Ubuntu has its own server dedicated platform, which is what I recommend be used for the application, which utilizes minimal machine resources. Ubuntu is widely used in corporate and server hosting environments which creates a solid foundation for community resources and update stability.
2. **Operating Systems Architectures**: At the core of Ubuntu, the Linux kernel is used to manage and control physical hardware/resources. Linux uses the monolithic kernel framework which means all services exist and execute within the kernel address space. Thus accessing the core OS architecture can be achieved through simple procedure calls. This kernel model provides a privileged position which aids in direct control over the operating system. Through these procedure calls, CPU scheduling, memory management, file management, etc., can be configured.

The Draw It or Lose It application would utilize a client-server type network architecture. The game and its resources would be centralized. Accessing resources or the game would be achieved through client requests and server responses (I.e. when a client connects to the webpage, a request is sent to the server, and the server responds with the appropriate resources).

1. **Storage Management**: The program and all its resources would be stored locally on a solid state drive. This solution fits perfectly for the size of users expected by The Gaming Room for their application. If more physical storage is needed, more local disks can be added as necessary (Yet this game is not large in size and will likely need little physical storage space).
2. **Memory Management**: The best solution for handling memory for this application would involve storing the array of all the unique pictures (200) in memory and using references to direct traffic to the correct unique image in memory. This would require 1.6GB of memory since each image is 8MB is size. This solution is superior to loading the same image multiple times in memory for each client (which could reach up to 8GB in memory usage). If memory of the system is limited in capacity, page filing can be used to offload images from physical memory to virtual memory.
3. **Distributed Systems and Networks**: REST will be used to as the primary method of communication between the server and web client. REST primarily uses HTTP which means any device capable of utilizing HTTP (most, if not all, operating systems) is capable of running the game application. REST is a stateless API which means all information regarding the client session is not stored on the server. The session related information is stored locally on the client, and sent to the server. Thus the server never loses track of where the client is within the application. This advantage should deal with small network blips/downtime.
4. **Security**: The data transferred between the client and server can be secured through HTTPS. REST is capable of utilizing HTTPS which encrypts data over the network. Likewise, REST has role-based authentication and authorization capabilities. OAuth 2.0 framework paired with an authentication foundation, such as Dropwizard’s Jersey authentication, is my recommended user authentication/authorization method. With this implementation, once a user is authenticated (via a username/password), a generated access token (OAuth 2.0) is sent back to the client and used in subsequent REST headers (until the token expires and login is required again) to authorize access rights/privileges to resources.

Another important security consideration is Ubuntu’s full disk encryption. This feature will ensure any sensitive information is secure locally on the server. This could slow performance of the application so EncFS is another option. EncFS is capable of encrypting certain files or entire directories. Thus sensitive-information containing directories pertaining to Draw It or Lose It could be encrypted.