

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 3.0

## Table of Contents

[**CS 230 Project Software Design Template** 1](#_heading=h.gjdgxs)

[**Table of Contents 2**](#_heading=h.30j0zll)

[**Document Revision History 2**](#_heading=h.3znysh7)

[**Executive Summary 3**](#_heading=h.2et92p0)

[**Requirements 3**](#_heading=h.tyjcwt)

[**Design Constraints 3**](#_heading=h.1t3h5sf)

[**System Architecture View 3**](#_heading=h.4d34og8)

[**Domain Model 3**](#_heading=h.2s8eyo1)

[**Evaluation 4**](#_heading=h.17dp8vu)

[**Recommendations 5**](#_heading=h.26in1rg)

## [Document Revision History](#_heading=h.lnxbz9)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 01/17/2023 | Lukas Pentowski | Initial version |
| 2.0 | 01/30/2023 | Lukas Pentowski | Evaluation completed |
| 3.0 | 02/17/2023 | Lukas Pentowski | Recommendations completed |

## [Executive Summary](#_heading=h.35nkun2)

The Gaming Room wants the development of a web-based version of the game Draw It or Lose It. Draw It or Lose It is loosely similar to the 1980s game *Win, Lose or Draw.* The web-based application will render images from a predefined library of drawn clues. There will be four rounds of play, each lasting one minute. Drawings will be rendered at a steady rate and be complete at the 30-second mark. If the initial team does not guess the answer the remaining teams will be able to have one guess with a 15-second time limit.

## Requirements

The current software requirements are:

* One or more teams able to join a game
* Multiple players assigned to each team
* Game and team names must be unique
* Users must be able to check whether a name is in use when choosing a team name
* Only one instance of the game can exist in memory at any given time.
  + This will be accomplished by unique identifiers for each instance of a game, team, or player

## [Design Constraints](#_heading=h.1ksv4uv)

* Compatibility between various platforms.

The client, The Game Room, has made the request that the web-based game must serve multiple platforms. These platforms have not been defined. For the purpose of this design constraint the various platforms will be defined as the computing platforms, i.e. operating system, and content distribution platforms, i.e. web browser. (Barick) According to StatCounter’s global stats, the top six web browsers were Google Chrome, Apple Safari, Microsoft Edge, Samsung Internet, Firefox, and Opera totaling approximately 96% of browser use. (StatCounter)

* Compatibility between desktop, tablet and mobile.

Though The Game Room’s original game, Draw It or Lose It, is currently available as an Android app, the porting of the game to a web-based application will allow mobile and tablet users to access the game without the Android app. The screen size will affect formatting of the UI. Also, the tablet and mobile device OS may affect the compatibility of the web-based application.

* Network capabilities of users.

The target user base is not centralized, therefore they do not have uniform Internet or phone data plans. Due to this the download and upload speed of the users will vary. Also, since the location of the user base is not centralized the server ping will vary from user to user. With a time sensitive game, having rounds of 1-minute and 15-second second chance guesses, ensuring the web application has low network demand is a must.

* Network capabilities of the cloud server.

Much like the user capabilities, the cloud server network must be able to handle the predicted traffic and support the upload/download requirements of the game.

* UI

The user interface (UI) must be cross compatible with the multiple platforms and devices used. This interface must also keep into account the various network capabilities of the user base.

* Project Timeline

No current target launch date has been announced. Due to this proper planning of the development of the project cannot begin.

* Budget

No current budgetary concerns have been announced. Due to this current planning of the development of the project will vary from the final plan. The Development Team may not be properly staffed.

* Use of a predefined drawing library

The Game Room uses a large library of stock drawings as clues instead of having a team member draw the prompt.

* Use of the Java programming language

Due to the request for a web-based application, Java was selected as the programming language.

## [System Architecture View](#_heading=h.44sinio)

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](#_heading=h.2jxsxqh)

There are seven classes as part of the program package for the “The Gaming Room UML Diagram”. These seven classes are the ProgramDriver, SingletonTester, Entity, Player, Team, Game, and GameService. The following is a description of the classes involved.

The ProgramDriver is where the public function main() is declared. This class item uses the SingletonTester class. The ProgramDriver is the class file that runs to execute the application.

The SingletonTester class that is used by the ProgramDriver contains one public function, testSingleton(). This function displays the id of the saved instances to verify the Singleton code is correctly working.

The Entity class contains the private attributes id, a long, and name, a string. It also contains the private method Entity(), as well as the public methods Entity(id, name), getId(), getName(), and toString().

The Player class inherits from the Entity class. The Player class contains the public methods Player(id, name) and toString(). This class is associated with the Team class and has the multiplicity of zero to many, meaning there is no limit on the number of Player class objects that can be associated with a Team class object.

The Team class also inherits from the Entity class. The Team class contains the private attribute players, which is a list item, as well as the public methods Team(id, name), addPlayer(name), and toString(). The Team class is associated to the Player class with a multiplicity of zero to many, as well as associated to the Game class with the same multiplicity.

The Game class inherits from the Entity class. The Game class contains the private attribute teams, a list, as well as the public methods Game(id, name), addTeam(name), toString(). The Game class is associated with the Team class with a multiplicity of zero to many, as well as the GameService class with a multiplicity of zero to many.

Lastly, the GameService class contains the private attributes games, a list, nexGameId, a long, nextPlayerId, a long, nextTeamId, a long, and service. The Game Service class contains a private method GameService(). The class also contains public methods getInstance(), addGame(name), getGame(id), getGame(name), getGameCount(), getNextPlayerId(), getNextTeamId(). The Game Service class is associated with the Game class with a multiplicity of zero to one.

The UML Diagram shows the use of encapsulation, with Entity, Game, Team, and GameService all having either private attributes or a private method. The use of encapsulation prevents those attributes and methods from being called outside of their respective class. This hides information and limits access to the state of the class object.

The UML Diagram also shows the use of inheritance, with the Player, Team, and Game class objects inheriting information from the Entity class. These classes inherit the id and name attributes from the parent class Entity. Using inheritance prevents having to repeatedly type the same code multiple times.

The UML Diagram demonstrates the use of abstraction in the code by showing the use of classes to represent the complex parts of the program, such as the GameService, Team, Player, and Game. The use of the variables id and name also is an example of abstraction. The use of inheritance prevents the need to rewrite all the attributes and methods from the parent class into the child class.

Lastly, the UML Diagram shows the use of polymorphism through the use of method overriding. The Entity class has a public method called toString(). Each of Entity’s child classes also have a public method called toString(). These child methods will override Entity’s toString() if called by a child object. Using the method overriding allows for using the same method name repeatedly, but having different outputs based on the class that used the method.

**"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](#_heading=h.z337ya)

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |

| **Server Side** | When hosting for developing applications for the Mac OS or iOS Systems, use of Mac is a great decision. (Horne, 2023) MacOS X Server support was discontinued in April of 2022 and is not compatible with any MacOS past Monterey. (Mac OS X Server, 2023) Instead the key features are bundled in every copy of the client MacOS. MacOS is a BSD based, Unix-like operating system, with a GUI (“Aqua”)Interface layered over their Darwin OS. (Architecture of macos, 2022) MacOS has a higher cost as a server due to the price of hardware. (Snell, 2016) Due to this, MacOS has been less used, leading it to having far less security risks than Windows. Though MacOS is not malware immune. (MacOS, 2023) | Lowest cost, due to being open source. (Horne, 2023) Linux is a Unix-like system that has several different distributions available, while also being able to create a new distribution. (Horne, 2023) With there being so many different distributions there may be issue with longevity. Support for the distribution can be rather short lived or last a long time. This does not mean a distribution cannot be used if support is discontinued. Red Hat Linux is one of the more popular distributions for web based hosting due to its strong security, but the support was discontinued in 2014. (Horne, 2023) Application licensing is based on what libraries and software components were used. (GNU General Public License, 2023)  Several bugs in the Linux kernel can lead to security vulnerabilities. There are frequent updates to address this issue, via Git. There are also several security modules to reduced kernel attacks. (Linux kernel, 2023) | Windows Server has an average of ten-year support longevity. This period includes five years of main support with five years of extended support. (Windows Server, 2023) Windows Server is more expensive than Linux to use due to the price of licensing. Any Windows product used will need additional licensing as well. (Horne, 2023) Uses SQL Database of MS SQL, instead of MySQL (Horne, 2023) Support for the Microsoft .NET framework. (Horne, 2023) | Use of several different applications have allowed mobile devices to fill the role of web server. (Mobile Web Server, 2022) The servlets available use Java API, Maven 2 Plugin, and Ant Tasks. (Mobile Web Server, 2022) These servers can handle lower user traffic and prevent the use of multitasking while running. (Gangwar, 2022) |
| --- | --- | --- | --- | --- |
| **Client Side** | Supports all web browsers but has been known to force third party browsers to act like Safari. Not all file types are supported. Though widely used, not as popular as Windows. Will take more time and expertise to properly support. | Supports all web browsers. Not all file types are supported. Not as popular in use as Windows or Mac. Will require more time and expertise to properly support. | Supports various web browsers. Most common system in use. High familiarity. Lowest time and expertise to support. | Due to different operating systems in mobile devices, must ensure a simplified application is provided for mobile support. Will take a moderate amount of time with multiple levels of expertise to support. |
| **Development Tools** | Java, Python, HTML, CSS, XML, SQL, SWIFT, JavaScript  Visual Studio Code/Eclipse | Java, Python, HTML, CSS, XML, SQL, JavaScript  Visual Studio Code/Eclipse | JavaScript, HTML, CSS, XML, SQL, Java, Python  Visual Studio/Eclipse | Java, Python, HTML, CSS, XML, SQL, SWIFT, JavaScript  Visual Studio Code/Visual Studio/Eclipse |

Recommendations

1. **Operating Platform**:

The Gaming Room should use a hosted public cloud-based server with Linux OS to host their web based application, Draw It or Lose It. As mentioned in the Evaluations section, the use of a Linux distribution for is the most cost effective, due to the GNU Licensing requirements and the operating system being open source.

CentOS and Ubuntu are the two most common Linux distributions in use for web applications. It is recommended to use Ubuntu for Draw It or Lose It. Though not as secure as CentOS, Ubuntu is still a secure stable build. (Tucakov, 2022) Ubuntu also is widely used in the cloud market and preferred for use in gaming servers. Ubuntu also has a large support community, a five-year support lifecycle for each Long-Term Support release, ability to install a graphical user interface and has an easy-to-use sudo command. (Tucakov, 2022)

1. **Operating Systems Architectures**:

Ubuntu is a Linux distribution built on Debian’s architecture and infrastructure. Linux is a monolithic kernel, meaning the operating system runs entirely in the kernel space, and is modular, allowing for inserting or removing loadable kernel modules (LKMs). (Linux Kernel, 2023) Linux uses a paged virtual memory for user memory management, while the kernel cannot be paged.

1. **Storage Management**:

The use of a cloud-based web server will allow for scalability of storage resources. There are two types of scalabilities: vertical and horizontal. Vertical (up and down) scaling involves changing physical components of the cloud server to increase or decrease the capabilities of the one machine. Horizontal (in and out) scaling involves adding or removing additional servers to meet the changing needs. (*What is cloud scalability?: Cloud scale* 2023) It is recommended to use horizontal scaling, along with a content delivery network, to meet the storage needs of The Gaming Room’s Draw It or Lose It.

Also, it is recommended to use sharding for the user database. This use of several servers will increase performance. (*Shard (database architecture)* 2023) This can also allow for regional information to be brought up for faster performance.  
  
The use of the cloud will also provide backing up storage due to host providers scheduling routine backups.

1. **Memory Management**:

The Linux OS uses a paged virtual memory system for improving memory space and utilization. In using a page virtual memory system, the operating system breaks down the information into pages and stores them in a designated section of the hard drive. That information is then swapped and pulled into the physical memory when needed to be executed. Some items are pinned and will not be placed into the hard drive. With using a paged virtual memory system it is important to keep swap times into account and hard drive transfer speed, to ensure total transfer time meets the “1 second” rule. (Nielson, 2009)

Java also utilizes a garbage collection service to free up memory by removing unused information. There are four different types of garbage collection services and the one that will be recommended to use is the Garbage First (G1) Garbage collector. This collector partitions code into multiple units and clears both young and old generations at the same time. It is recommended to be used for user-based applications due to the low rate of freezing of the application. (Knupfer, n.d)

The use of a Content Delivery Network will also assist in reducing load times by having assets spread across servers that will be located closer to the various regions.

1. **Distributed Systems and Networks**:

The use of RESTful APIs allows for stateless, cacheable, layered, client-server architecture.

Statelessness reduces server load by reducing information sent in communication between client and server. Each packet of information does not need to reference any that came before. Cacheability can improve scalability and performance by reducing the communications needed between the client and the server. A content delivery network can store the cache, as well as the client’s memory. The use of a layered system allows for scalability by having load balancing and shared caches. (*Representational state transfer* 2023)

Load balancing directs traffic to different servers based on different approaches (Round Robin, Least Connections, Least Time etc.) This improves response times. Load balancers also continuously check the servers, ensuring that traffic is only directed to online servers. (*Load balancing approach in Distributed System* 2022) This improves overall reliability. Load balancing can also improve security through DDoS protection, further improving reliability.

1. **Security**:

In order to ensure security, it is recommended to use multifactor authentication for ADMIN level users, expiring passwords, lowest level of privileges per account level, Attribute-Based Authorization, input validation, encryption, SSL, and logging. (Jones, 2022)

Public Cloud Service Providers maintain up-to-date software patches, security updates and use of firewalls. Security features are also tested more, due to the historical higher rate of attacks on public CSPs. There is also more capital available to be used, due to the shared costs of clients, to afford security upgrades. The use of a CDN can help reduce the impact of a distributed denial of service attack.

Java contains several different APIs, tools, and common security algorithms inherent to the language. Java also uses garbage collection, which reduces vulnerabilities caused by memory leaks.

References

Gangwar, M. (2022, August 3). *How to host a web server on Android*. DigitalOcean. Retrieved from https://www.digitalocean.com/community/tutorials/host-a-web-server-on-android

GeeksforGeeks. (2022, June 8). *Load balancing approach in Distributed System*. GeeksforGeeks. Retrieved from https://www.geeksforgeeks.org/load-balancing-approach-in-distributed-system/

Google. (n.d.). *Overview of Cloud Game Infrastructure  |  cloud architecture center  |  google cloud*. Google. Retrieved from https://cloud.google.com/architecture/cloud-game-infrastructure

Horne, K. (2023, January 3). *Operating Systems and web hosting: Read here before making your choices*. Digital.com. Retrieved from https://digital.com/best-web-hosting/operating-systems/

Jones, J. (2022, February 8). *11 best practices for developing secure web applications*. LRS Web Solutions. Retrieved from https://www.lrswebsolutions.com/Blog/Posts/32/Website-Security/11-Best-Practices-for-Developing-Secure-Web-Applications/blog-post/

Knupfer, Franz(n.d.). *Java garbage collection: What is it and how does it work?* New Relic. Retrieved from https://newrelic.com/blog/best-practices/java-garbage-collection

Neilson, J. (2009, October 4). *Powers of 10: Time Scales in user experience*. Nielsen Norman Group. Retrieved from https://www.nngroup.com/articles/powers-of-10-time-scales-in-ux/

Simic, S. (2022, December 5). *LAMP stack - what is it, Advantages & Alternatives: Phoenixnap KB*. Knowledge Base by phoenixNAP. Retrieved from https://phoenixnap.com/kb/what-is-a-lamp-stack

Snell, J. (2016, January 20). *4 reasons why you should run your own mac server, and 4 reasons why you shouldn't*. Macworld. Retrieved from https://www.macworld.com/article/227231/4-reasons-why-you-should-run-your-own-mac-server-and-4-reasons-why-you-shouldnt.html

StatCounter. (n.d.). *Browser market share worldwide*. StatCounter Global Stats. Retrieved January 17, 2023, from <https://gs.statcounter.com/browser-market-share>

Tucakov, D. (2022, November 14). *Centos vs ubuntu: Choose the best OS for your web server*. phoenixNAP Blog. Retrieved from https://phoenixnap.com/blog/centos-vs-ubuntu

*What is cloud scalability?: Cloud scale*. VMware. (2023, February 17). Retrieved from https://www.vmware.com/topics/glossary/content/cloud-scalability.html

Wikimedia Foundation. (2022, December 6). *Mobile web server*. Wikipedia. Retrieved from https://en.wikipedia.org/wiki/Mobile\_Web\_Server

Wikimedia Foundation. (2022, October 11). *Architecture of macos*. Wikipedia. Retrieved from https://en.wikipedia.org/wiki/Architecture\_of\_macOS

Wikimedia Foundation. (2023, January 12). Linux kernel. Wikipedia. Retrieved from <https://en.wikipedia.org/wiki/Linux_kernel#Architecture_and_features>

Wikimedia Foundation. (2023, January 13). *Windows server*. Wikipedia. Retrieved from https://en.wikipedia.org/wiki/Windows\_Server

Wikimedia Foundation. (2023, January 26). *GNU general public license*. Wikipedia. Retrieved from https://en.wikipedia.org/wiki/GNU\_General\_Public\_License

Wikimedia Foundation. (2023, January 31). *MacOS*. Wikipedia. Retrieved from https://en.wikipedia.org/wiki/MacOS

Wikimedia Foundation. (2023, February 5). *Mac OS X Server*. Wikipedia. Retrieved from https://en.wikipedia.org/wiki/Mac\_OS\_X\_Server

Wikimedia Foundation. (2023, February 18). *Shard (database architecture)*. Wikipedia. Retrieved from https://en.wikipedia.org/wiki/Shard\_(database\_architecture)

Wikimedia Foundation. (2023, February 19). *Representational state transfer*. Wikipedia. Retrieved from https://en.wikipedia.org/wiki/Representational\_state\_transfer