

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template** 1](#_gjdgxs)

[**Table of Contents 2**](#_30j0zll)

[**Document Revision History 2**](#_3znysh7)

[**Executive Summary 3**](#_2et92p0)

[**Requirements 3**](#_tyjcwt)

[**Design Constraints 3**](#_1t3h5sf)

[**System Architecture View 3**](#_4d34og8)

[**Domain Model 3**](#_2s8eyo1)

[**Evaluation 4**](#_17dp8vu)

[**Recommendations 5**](#_26in1rg)

## [Document Revision History](#_lnxbz9)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | Jan 22, 2023 | Matthew Courts | Initial Development design |

**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](#_35nkun2)

The Gaming Room is looking to develop a new web-based called “Draw It” or “Lose It”, based off a TV show from the 80’s. The grame is built around teams of players competing against each other in a round at a time. There are four rounds, with the goal of each to solve a puzzle with the use of images.

## Requirements

*Game is intended to be browser based. For this a server client setup will be used. The pictures and the game can be served on the server leaving the client with minimal load times based on their internet connection speed.*

*Browser based design will account for wide variety of OS as the client will utilize a web browser. Web Browser display data on using HTML, CSS, Javascript or applets provided in the JAVA or ASP.Net, there is also a standard format for how data is transmitted and handled, which is taken care of by the browser.*

*A domain and server will need to be set up along with proper routing for web browsers to access the game service.*

## [Design Constraints](#_1ksv4uv)

* Game needs to be separated into a client-server structure
* Client needs to be designed to run on multiple platforms
* Players will need to be able to connect the client to the server to find a team
* Once a team is formed service will need to pair the team with another full team
* Service will operate the game, controlling the round counter, gameinstance, serve the images and set the puzzle
* Service will need to host the images and have them accessible to the client.
* Drawings will be made visible at the 30 second mark
* If the team fails to guess the puzzle, the opposition team will have 15 seconds
* Client-Server structure should aim for real time updates given timing mechanism.

## [System Architecture View](#_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](#_2jxsxqh)

Entity will be the base class that houses the common data inherited by the Game, Team and Player classes. Game service will house zero to infinite amounts of games, where a game will contain zero to many teams. Teams will contain players to an infinite amount. Each class above needs an ID and name, this data structure is provided by the entity class along with accesssor functions. Game service is a singleton allowing only one instance of it. This will be the main service hosting all the games, containing the functions to manage the hosted games.

**"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](#_z337ya)

* **Evaluate various platforms for their characteristics, advantages, and weaknesses for hosting a web-based software application.** Consider the following in your evaluation and articulate your findings in the software design template:
  + Does each of the operating platforms offer a server-based deployment method where the website will be hosted?
  + What are the potential licensing costs to the client, The Gaming Room, for the server operating system?

**Client Side:** The client wishes to move beyond their current Android-only application to supporting players on iOS and Android mobile platforms, as well as traditional desktop-based operating systems. The application must be delivered as a modern, responsive HTML interface running inside the web browser for desktop clients (Linux, Mac, Windows), as well as on mobile platforms. Each will be capable of communicating with the back-end web application running on the server.

* **Determine the software development considerations (cost, time, expertise) that are necessary for supporting multiple types of clients.** Consider the following in your evaluation and articulate your findings in the software design template:
  + What is required of the application development process to ensure the application is compatible with all web browser platforms and mobile devices?

**Development Tools**

* **Identify the relevant programming languages and tools (IDEs and other tools) that are used to build this type of software for deploying on each operating platform.** Consider the following and articulate your findings in the software design template:
  + What impact do these technical requirements have on a development team? Consider whether multiple development teams may be needed.
  + Are there licensing costs related to the development tools?

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Accessible documentation and a flexible OS that allows for a consistent environment to be set up for the hosted service. Mac will have OS and possible software licensing for hosting and development. | Accessible documentation and a flexible OS allow for a consistent environment to be set up for the hosted service.  It is commonly used as a web service OS, with very flexible development and hosting options. no licensing costs only the hardware usage. | Will require a bit more effort to integrate and service, set up a firewall, and routing.  Forced OS updates may require a long-term maintenance plan. Licensing costs will have to be considered and maintained. | Poor hardware for Server hosting. Mobile focuses on low-power applications often with lower amounts of system resources.  Portable and can be run almost anywhere. |
| **Client Side** | For a web-based client, the development will need to develop for web browser compatibility. Most commonly used browsers follow standards for reading websites  OS has access to most common web browsers | For a web-based client, the development will need to develop for web browser compatibility. Most commonly used browsers follow standards for reading websites  OS has access to most common web browsers | For a web-based client, the development will need to develop for web browser compatibility. Most commonly used browsers follow standards for reading websites  Easiest target to develop for as this is the most used consumer OS. | Web Browser support may be limited or require more effort to design a mobile client interface.  Often times this effort is put into a client app designed to run on the mobile OS and not in the browser. This would require specific development for different mobile OS and specialized developers. |
| **Development Tools** | Common languages are available with most open-source IDEs and text editors are available. Professional IDEs are available allowing a broader range of available languages. Mac and Linux allow for replication of the server environment.  Front-end development will be using HTML/CSS/Javascript  Free tools can be used to develop, Mac has paid tools available for their ecosystem.  Has the flexibility of replicating Linux, Mac, and Windows client environments for servers and clients with Virtual Machines. | Common languages are available with most open-source IDEs and text editors are available. Professional IDEs are available allowing a broader range of available languages. Mac and Linux allow for replication of the server environment.  Front-end development will be using HTML/CSS/Javascript  Free tools can be used to develop, and paid tools are available.  Has the flexibility of replicating Linux, Mac, and Windows client environments for servers and clients with Virtual Machines. | Most IDEs are targeted to run on windows, along with targeting the production environment to be windows as well. More text editors are available.  Replicating the server environment may take more setup and configuration or the use of a VM  Free tools can be used to develop, and paid tools are available.  Has the flexibility of replicating windows and Linux client environments for server and client.  Mac can be replicated in Virtual Machines. | Development on a Mobile platform is not well received nor recommended. Notebook-style laptops either come in windows or Google OS, with limited resources or app access. |

## 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 operating platform will be a common web browser, which will run on common operating systems allowing a wide range of user access. Web browsers to focus on Will be Chromium(Google Chrome), Firefox, Opera, Edge, and Safari, as these read the HTML, CSS, Javascript, and WebAPI standards allowing us to design to the standards first.
2. **Operating Systems Architectures**: The service is written in java allowing for it to be run on any server OS with the JRE environment set up. this will allow flexibility in determining server costs, the recommended is Linux and this has only hardware, and administrative costs. This system is flexible and can be run in a cloud environment allowing for future scalability.
3. **Storage Management**: Data storage can be used in different formats from text files to cloud bucket servers. Security should be considered no matter the data collected and stored. Databases can also be utilized as a storage medium along with a public file share for common public-facing assets. For The Gaming Room, the recommendation is to use a static public file share, storing while storing the paths in a database that will already be used to store player information. This will allow for expanding game assets later on, along with a database setup for future development.
4. **Memory Management**: memory management can be handled during development, most programming languages have garbage collection built in releasing memory when no longer needed, however with some languages this needs to be manually tracked and tested. Java with the current development plan does contain garbage collection but tests should be done and software should be developed with memory usage in mind. Monitoring app load during peak activity can determine memory requirements. This can also reduce server memory costs, with the use of cloud infrastructure as you can customize accessible memory.
5. **Distributed Systems and Networks**: With a Client-Server setup the server can be scaled by placing a router in front of the services. These services will also need to share a common database and file share for the images and other static resources. The database can also be scaled based on a number of service instances with a master and replicated configuration. The scaling setup can allow for more parallel setups to be implemented. Each instance of a service can be used to provide connections to different areas in a country or even different countries. Allowing to control server costs during game activity spikes and slow periods.
6. **Security**: Security should be developed from the start, and data access should be restricted to only those with access. User management can be set up to allow only specific users access. Password management should follow common-day practices for users and game owners. The services can be isolated using container technology separating data streams from service architecture. Data streams should be encrypted and handled on a different container only allowing authorized access from the other containers in the app.