

# **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 | 07/12/2021 | James Roberts | Talk about problems and ideas before the project gets started. |
| 1.01 | 07/26/2021 | James Roberts | Added information to the Evaluation section. |
| 1.02 | 08/10/2021 | James Roberts | Added information to the recommendation section. |

## [Executive Summary](#_sbfa50wo7nsh)

Draw it or lose it is an Android game that will be developed as a web-based game. The basics of the game is to pull random images for the already extensive library to display from a blank screen to a complete picture in the span of thirty seconds. If the team does not guess the image correctly in their allotted time, then another team will have a chance to guess, however they will only have fifteen seconds to guess correctly. There will be multiple teams with multiple players on each team. Each game and player will have unique identifiers to make sure that name is not already in use.

## [Design Constraints](#_2et92p0)

Trying to match the looks and button layout of the Android app to a web-based app will be challenging.

Upkeep and adding to the games will need two team of developers because the coding languages will be different.

Making sure the game works on all web browsers.

## [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 UML diagram below shows how the GameService section has all of the information to get game id’s, team ids, and player id’s. Here you will also be able add a new game, team, or player. All this information will be collected by the Game section. In the Game section the teams will be assigned, and players will be added to teams. This will also be the place to pick team names. All this information will be shared with the Entity section and Team section. The Team section will be shared with the Game section and the Player section. This is where their players will be gathered and sorted into groups. The Player section will be where players will be able to create new ids. The Game, Player, and Team sections will al be connected to the Entity section where the ids and names will be checked will be sent to the game to play.

****

## [Evaluation](#_2o15spng8stw)

This game can be hosted on a distributed server that will allow the server to store all of the information for the game and let multiple web browsers access the information. This will also let the users interact with the game. With this being a web-based application, any modern mobile platform, desktop, laptop, or tablet will have access to the website. This will grant anyone access to the game through a web browser.

Every operating platform will have a server-based deployment where the website will be hosted. To reduce the cost of expanding this game if a Linux based server is used, because it is open sourced, the cost will be free in this aspect of the process. Backend development using the REST API will allow this game to be scaled as big as it needs to be with minimal additional effort once the database is already built.

To keep everything as streamline as possible, building the game in HTML, CSS, and JavaScript there will only need to be a single source code that can be published on a web browser. This will keep the code easy to modify and update as needed. To create this code the Visual Studio Code would be a great choice of IDE. It is also free. So, the only cost will be running the server and the cost of developing, maintaining, and updating the game.

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Lots of terminal commands to set up, access, and change the server. Simple to navigate. | Lots of terminal commands to set up, access, and change the server. Difficult o navigate. | Has the most software available. Simple to navigate. | Really Difficult and improbable. Hard to use. |
| **Client Side** | Cost a little more than windows  Harder to set up than the other three. Takes longer to set up than Windows or Mobil. | Lowest cost. Hardest to set up and takes the longest to set up. | Average cost. Simplest to set up and least amount of time. | Average cost. Slightly more complicated to set up than windows. Gives ability to check and receive updates from any place. |
| **Development Tools** | HTML/CSS/JavaScript, python, Java, Php, Ruby on Rails, React Native, C, C++. Whatever ide the developer prefers. | HTML/CSS/JavaScript, python, Java, Php, Ruby on Rails, React Native, C, C++. Whatever ide the developer prefers. | HTML/CSS/JavaScript, python, Java, Php, Ruby on Rails, React Native, C, C++. Whatever ide the developer prefers. | HTML/CSS/JavaScript, python, Java, Php, Ruby on Rails, React Native, C, C++. Whatever ide the developer prefers. |

## 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 using the Windows operating system because it has lots of support and is the most popular operating system in the world.
2. **Operating Systems Architectures**: Windows comes with memory and file management. It can also be used to write code in any language needed.
3. **Storage Management**: With Windows file management the storage is simple and quick because everything can be saved directly to the machine.
4. **Memory Management**: With up to four gigabytes of memory, Windows will be able to store everything that needs
5. **Distributed Systems and Networks**: Windows servers talk well together and can be combined to have even more computing power.
6. **Security**: Windows currently has the basics of security, so adding security when available will be the best approach.