A video game controller and a cable

AI-generated content may be incorrect.

The Gaming Room

# CS 230 Project Software Design Template

Version 1.1

## Table of Contents

[CS 230 Project Software Design Template 1](#_Toc7193)

[Table of Contents 2](#_Toc7194)

[Document Revision History 2](#_Toc7195)

[Executive Summary 3](#_Toc7196)

[Requirements 3](#_Toc7197)

[Design Constraints 3](#_Toc7198)

[System Architecture View 3](#_Toc7199)

[Domain Model 3](#_Toc7200)

[Evaluation 4](#_Toc7201)

**Recommendations**

## Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| 1.0 | 05/21/2025 | Oumar Kenneh | Changes were made to the cover page contents, the document revision history, the executive summary from before, the design constraints from before, the system architecture view, domain model and recommendation. |
| 1.1 | 06/28/2025 | Oumar Kenneh | Updated evaluation and requirements, looked over previous contents |
| 1.2 | 06/06/2025 | Oumar Kenneh | Updated Recommendations section |

**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

## The client wants us to create a cross-platform web game from the game Draw It or Lose it. However, otherwise, it won’t be quite the same as Draw It or Lose It as that first game is only on Android. The concept of the game is to have multiple teams of three or more people going four rounds that are each one minute long. Each image will be drawn from a repository of stock photos and presented in succession. If the present team fails to respond, any of the other teams can have one guess each within a time limit of 15 seconds.

## Requirements

## The client would like the game to be made under the pretext of a few requirements:

## • True Multi Team support with Multi Skins (Each team can have multiple players).

## • There should be only one instance of the game at a time

## • Team name selection - check if a name is taken

## • Unique ID for each game, team and players and Unique Names

## Design Constraints

## There are several requirements for the game application to function well:

## Cross-platform: The game was originally made for android, and now it needs to be transported onto other platforms. This will be achieved by converting the app to a web-app which can be used on other platforms. We can communicate using HTTP over REST API, without needing to select various capable languages.

## User Interface- The game would have to be almost identical to that of the android app, or it will have to take a completely different turn like creating a whole new game design.

## Many Teams/ Many Players: The game should allow multiple teams, and each team would need multiple players. In order to achieve this, you will have to use a client-server setup to ensure that the server can occupy more than 1 player at once.

## Unique IDs: Game and team names are unique, and only one game is currently in memory at a given time. To get it working may require managing memory better because we have a new game to export it to so it can run elsewhere. We will also need to find the player platform ids.

## Images and Copyrights: The Gaming Room wishes to use your images for a game called Draw It or Lose It. We'll be using those images that you use in the android system, and we'll ensure they're compatible with every platform. We’ll also need to ensure license for images and rights to reproduction of copyright images are in place during design.

## System Architecture View

## Note: There is nothing required here for these projects, but this section is here to remind you to describe the system and subsystem architecture i.e. physical tiers and components if applicable are required for other projects. It is also necessary to describe a logical topology of the communication and storage aspects to have a full picture of the architecture.

## Domain Model

## Entity Links Game, Team & Player Class. What this means is that they all receive or inherit information from Entity. We can demonstrate this in UML with inheritance. Making Entity a superclass. When we look at their relationship, we see Team and Player goes as a “has a” type. So, Game has a team and GameService has Games. In UML, if we use it, we say it as aggregation (HAS-A). And when a user ‘’ has a I mean it is an instance of a class and has reference to an instance of another class. Now if we back up and examine this diagram, we see GameService has a reference of Games, Games a reference of Teams, and Team a reference of Players.

.

A diagram of a computer program

AI-generated content may be incorrect.

## Evaluation

Leverage the information you have about the features both good and bad of the kinds of platforms (Linux, mac, windows, mobile) below and summarize them all. And as you finish the table, you’ll keep in mind your client’s needs and vision and think in terms of the whole, because it has to all work together.

In Each Cell, delete the brackets and type your own prewriting covering the information requested.

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Server Side**  **yes** | Mac has easy accessibility — it can run on numerous platforms applications with ease and through server-side configuration. Simple to use GUI. Flexible terminal commands. It also can face hardware limitations seen when compared to a upgradable Linux box or a Windows pc. | Cost friendly. The platform is hard to use. An easy to use server setup and access command shell. Variety of distribution options available, NOT locked sourced. Known for its versatility in servers and embedded solutions, because of their high modifiability. Linux is also relatively more secure than Windows or MacOS. There aren’t very many options for prebuilt machines, so one generally has to custom-build it, which is more expensive when you can’t buy those parts wholesale. The files aren't compatible! | Server side is expensive. User friendly GUI. Has a command prompt. Windows authenticated interface with corporate, Active Directory based servers for corporations, straightway. Deficient in certain aspects, like security policies and mobility planning. | The specs are better on other devices. The specifications of a mobile device depend on the device type (i.e. the device model). For mobile applications the requirement is usually to be light and consume less battery. There are some cheap server apps around. Mobile devices are typically more susceptible due to as as security is concerned than Macs and PCs. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Client Side**  **yes** | Moderate in time and expertise. They require very precise skills for OS, but once you get the skills down it is very easy to use. Its hi cost because to develop for Mac or iPhone is a Mac/MacBook | It is a lot of expertise and a lot of time. You need something with a Linux file-system on it for the OS to run. Linux boxes are very inexpensive, and tons of control | Pricier than Linux systems, however, there are various levels of Windows that are less expensive based on how you use your OS. Very easy to learn and understand how to support a | Mobile “development infrastructure leaves a lot to be desired with a lot of apps and tooling developed but not as easily accessible/feature rich as PC/Mac.” Many different OS and devices can |

### 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**:

Windows is the most suitable operating platform through which The Gaming Room can take Draw It or Lose It to other computing devices. Ideally Windows as you won't be starved for IDEs to use and I find it's straightforward to use the GUI and code efficiently. Windows also is completely compatible with cross-platform play in many of today's gaming titles. For games I think windows is a good choice for developers and consumers overall.

1. **Operating Systems Architectures**:

Microsoft Windows Microsoft Windows is a graphical based operating system developed and marketed by Microsoft Inc.). The Windows OS architecture is a layered model and is split into the following main components: Kernel, Executive, Subsystems, User-mode components, Hardware abstraction layer. The kernel is the central part of your OS that's responsible for allocating resources and defines how tasks and applications can interact with hardware. The executive layer is comprised of high-level system services, such as memory management. The sub-systems are functioning components for realizing specific functions such as executing win32 application. The user-mode portion of the driver stack consists of Windows Shell and other system services and drivers. It is a layer in kernel where it provides an interface to interact with the hardware devices in such a way that the low-level details are kept abstracted.

1. **Storage Management**:

Let’s kick off with Windows 10 which shares this brilliant feature called storage sense – it’s essentially an assistant who works directly with OneDrive and automatically frees up space by making locally available files online-only. It’s essentially a sort of floppy disk cleanup in automation. You have the freedom to free up manually by selecting the items to safely remove anything that Storage sense lists.

1. **Memory Management**: Windows 10 storage sense would enable storage and management of Draw It or Lose It photos and game players. It also lets you store them together in a single safe location in memory.
2. **Distributed Systems and Networks**:

Networked multi-user interaction systems, such as network games generally have a database shared between physically distributed users who interact over the network. Network game developers today have to code the shared world database and inter-player communication from the ground up.

1. **Security**:

Authentication and authorization is the foundation of secure software on any platform. Username and password is the right way to authenticate for this app. Encryption is crucially important, the user password stored in the database should be in a hashed form and not in plaintext to be available for viewing. Some of the other ways we as developers can protect the users is secure coding, as well as input and error validation and regularly maintained updates and patches. Windows provides free security protection that includes an antivirus program called Microsoft Defender. Based on previous test, it has up to 100% real time protection rates. It guards against malware or viruses that could cause harm to the device. You would also have to consider Database security and other servers where your application has to be secured like DDoS protecting server etc.