

The Gaming Room

# **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 | 04/1/21 | Garet Kean | Recommended changes and evaluation |

**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 wants to expand their web-based game “Draw it or Lose it” to run on multiple platforms. Currently the game does run on Android. The game involves multiple teams with multiple players on each team with four rounds lasting one minute each. A picture is selected at random and slowly shown to the teams giving them one minute to guess the picture, if no one can guess the picture then each of the opposing team members will have 15 seconds to guess what the picture is.

## [Design Constraints](#_2et92p0)

Some design constraint include the number of teams required to play, the number of people required for each team, and expanding the game to multiple platforms will require using similar languages and possibly integrating the existing code.

## [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)

Game, Team, and Player have an “is-a” relationship with Entity because they are all inherited from the Entity class. Since all three (Game, Team, and Player) have common attributes together they are inherited from Entity, which holds all of the common attributes, making Entity a Super Class.

The relationship between team and player is “has-a” as in team has a player. Game and team relationship is also a “has-a” relation. Has-a relationship means that one class has a reference to an instance of another class. In this UML diagram GameService can have multiple instances of Game, Game can have multiple instances of Team. And team can have multiple instances of players.

****

## [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 OS is available on apples website ([www.Apple.com](http://www.Apple.com)) and very affordable to deploy. | Linux has multiple distributions for server handling, is low cost, and open source. There are less linux users with much experience than there are for other operating platforms so finding someone with good experience with running linux servers will be essential. | Windows provides servers at [www.microsoft.com](http://www.microsoft.com). Windows comes fully functional and is the most commonly used operating system. Windows will be much more expensive but will offer more users with experience using windows on a server. | Servers should not be hosted from mobile devices and will require stationary servers. If a mobile server is started it will gives the user to operate the server on the go. |
| **Client Side** | Mac OS is not open source so the cost and time to build will be similar to that of a windows setup and will heavily rely on the expertise of the individual doing the set up. | A high level of expertise is required to operate with Linux and they would need enough time to work with because Linux can be complicated. The cost should be low considering that Linux is open sourced and not widely used. | The cost of Windows will be more because Windows is not open source. Time will depend on the experience level of users, but finding users who have experience operating windows servers will be easier to do. | Cost of mobile devices will depend on the mobile carrier. Minimal expertise will be required but more time would be required for multiple operating platforms and devices. |
| **Development Tools** | Swift is a good option to use for Mac. Mac can run all languages as needed. | Linux will work with Eclipse and visual Studio, giving developers a familiar IDE to work with. | Windows will also work with Eclipse and Visual Studios to keep things simple. | Multiple languages can be used as needed for mobile devices including swift, java, python, and Ruby. |

## 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 recommend starting with Windows for ease of use with IDE’s and the larger software availability. Having Windows will require a minimal expertise to operate and a lower cost to get started.
2. **Operating Systems Architectures**: Windows provides the services needed for windows applications, having stationary servers with windows will ensure easier operation.
3. **Storage Management**: Storing files on a hard-drive is a good option for stationary servers. I recommend using a cloud based storage to save space. Windows makes managing and creating files easy.
4. **Memory Management**: Draw it or Lose it will require a lot of pictures to be saved and can easily be saved in specific folders on a hard drive with windows or stored in a cloud.
5. **Distributed Systems and Networks**: Using Unity on windows will link services together from multiple platforms and minimize dependencies. To prevent outages, a larger server base can be built along with backup power supplies.
6. **Security**: Windows comes with built in malware protection that operates at all times and schedules scans as needed. It is not recommended to rely on Windows for personal data protection including e-mail address, passwords, etc.