

Draw it or lose it

# **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 | <08/01/21> | Howard Gimore | Added Server-client and dev tool explanations |

**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)

Draw it or Lose it an Android based application developed by The Gaming Room would like to branch their game into cross-platform environment. The game is a multiplayer team based objective game that has multiple game modes and multiple teams. In order to have the game function correctly the gaming room needs an environment that will utilize their library of drawings uniquely distinguishing each team.

## [Design Constraints](#_2et92p0)

1. Multiple platforms will have different toolkits required to ensure proper cross-play and functionality can occur.
2. Each game will need a unique ID generated to allow for memory saving
3. Different API’s will be required similar to the first constraint

## [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)

ProgramDriver is the main() containing the information. ProgramDriver will talk to SingletonTester to determine if GameService is already utilizing a game ID that is requested. The parent class “Entity” holds Game, Team, and Player classes. Game, Team, and Player. The child classes all inherit Entity. A Player cannot have a Team, but a Team can have a Player. The classes act to prevent misinformation from being inputted for example a square can be rectangle but a rectangle cannot be a square. No duplicate players can be on a team and no team can have duplicate names since there all assigned a unique ID.

****

## [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 OSX server is a pay to use server, Apple.com references that it is inexpensive to implement. This being said cheap doesn’t mean that it’s the best and apples servers are not as popular as its equals. | Linux servers being the most widely used due to its historic age and security is low cost and open source. Not many users are technically savvy in Linux however, MAC os uses an integrated version of Linux therefore anyone who can operate and program with MAC is well equipped to dive into Linux. | Windows utilizes its windows server. Windows server is widespread and easy yet costly to implement. Since its using windows as its base it would be easy to find windows users to operate and maintain it. | Do not recommend running servers on mobile devices, though least cost effective from the other choices but due to lack of power and capabilities its it not recommended. |
| **Client Side** | Cost would be similar to windows since expertise is depended on the time spent on the Mac OS. Someone with a lot of experience with MAC will have a faster completion time. | Linux is open source so cost shouldn’t be a factor, Linux is not commonly used but is related to Mac OS which could enable it to be used by someone who is apt with mac and linux. | Cost would be similar to mac since expertise is depended on the time spent on the Windows. Someone with a lot of experience with windows will have a faster completion time. | Cost is not an issue, There are multiple operating systems however the two dominants are iOS and Android OS and its derivatives. It would be easier to use due to this. |
| **Development Tools** | Swift is the mose used dev tool on Apple however some Microsoft products have been used as well like visual studio. | Eclipse is used on linux as well as Atom, due to there exportability between OS it makes it easier for java developers and C+ users to communicate with the IDE. | Eclipse and Visual studio are the primary IDEs for windows, along with Visual Studio Code. They are very versatile and an IDE like VSCODE allows for multiple language librarys to be used and downloaded with ease. | Iphones use swift primarily from Mac os to program due to the apple ecosystem however IDE’s like visual studio and eclipse allow functionality to program and test Android and Iphone OS just not as efficient if it was within the ecosystem. |

## Recommendations

Analyze the characteristics of and techniques specific to various systems architectures and make a recommendation to The Gaming Room. Specifically, address the following:

**Operating Platform**: Window would be the most cost effective and appropriate operating platform that will allow The Gaming Room to expand Draw It or Lose It to other computing environments.

**Operating Systems Architectures**: Windows offers 2 modes for development and security, User mode (the general mode that we use on day to day) and Kernel mode(uses encryption to project the data being utilized also requires hard connection rather then wireless.)

**Storage Management**: SSD will always prevail over hard disk due to threading and transfer speeds as well as ease of real estate management.

**Memory Management**: Windows uses paged allocation to separate old memory from newer memory into fixed frame then utilizes virtual memory space to frame data into smaller sizes.

**Distributed Systems and Networks**: Distributing the application is the best way to progress the game. Connecting different servers to decrease the load and increase efficiency during high volume of traffic connecting. Load balancing will be used for this approach and load balancers will increase the response time too.

**Security**: Security is top priority, SSL (Secure Socket Layer) and a form of cloud security will help prevent against game hackers that ruin the experience and data breaches.