

<Name-of-Software-Application>

# **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 | <5/30/2021> | <Oliver Milani> | <Explained new software design additions. > |

**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 is looking to move their successful Android game, Draw it or Lose it, to a web-based-game.

## [Design Constraints](#_2et92p0)

The Constraints below are what the company is looking for.

* The game can only run one instance.
* It must be able to have multiple teams with multiple players.
* These teams and game names must check for uniqueness.
* Must be able to run on multiple platforms.

These constraints should be simple to add, with any code converting being the most time consuming task. Android apps are often in Java which will help with transferring 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)

At the top we see the Entity Class which is the super. It is inherited by the Player, Game, and Team classes. At the bottom from left to right we have the GameService class that has a reference to the Game class. That Game class has a reference to the Team class, which has a reference to the Player class. At the top left is the SingletonTester and the ProgramDriver class. The SingletonTester is used by the ProgramDriver. The ProgramDriver class is what executes the application.

****

## [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** | Server configuration is well managed. Terminal commands allow for good flexibility. | Has great server config and is not a weak OS like it once was. It can do most things Windows can for this type of application. | Easiest to use. Has a ton of software and support. | Generally not ideal. Needs to be sourced from a desktop. |
| **Client Side** | A good amount of time and expertise will be required. Cost will be decent. Integration with other platforms will be ok. | Should be similar to Windows, just a bit trickier. Integration with other platforms may be a bit more difficult. | Will integrate well with other devices. Easy to use and very little time cost. Less cost in general. | Easy for clients to download, play, and update. Cross play with other devices will be difficult. |
| **Development Tools** | For Mac we will be using swift. It will be able to support many scripts and a lot of programming languages like Java, Python, and Ruby. | Similar to windows here with eclipse and visual studio being available. It will be able to use the same code to allow for seamless integration. | Run through eclipse or visual studio. Will be able to use almost all languages with java or javascript being the most important ones for this application. | Programmed through another OS and will be using Java as its code. |

## 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 first operating platform for The Gaming Room to start with is Windows. It will allow for a good jumping off point as it will be easier to start off with and make integrating other platforms easier as well.
2. **Operating Systems Architectures**: Windows provides a lot of tools to improve the application. The access you have to the kernel architecture will help make sure the game runs as smooth as possible. This can also then translate to the mobile application.
3. **Storage Management**: Storage management is also very easy. Windows provides ways to organize and manage files on your hard drives. It also gives access to a very strong cloud data service. There will be plenty of room for large projects and many ways to manage them.
4. **Memory Management**: The programming languages and IDEs give you the ability to code in a way that is less memory intensive. You never want to leave a group of users out due to lack of memory and there are a lot of ways you can reduce memory cost.
5. **Distributed Systems and Networks**: Using Java will be the best way to communicate between various platforms in my opinion. All operating systems can use Java and it is some of the strongest code to use. If the network is being run from a central location, it can allow for each operating system to connect to it and distribute data back and forth without issues. This will then need certain redundancies for power outages and extensive testing for max player numbers per server.
6. **Security**: Keeping customer data safe is important. We will need to use strong software to protect from those seeking user information. An anti-virus and anti-cheat scan will be performed at the start up of the application for the user. Continuous checks on the server side will be happening always.