

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 | 09/21/2022 | Thomas Dang | Initial summary and evaluation |
| 1.1 | 10/04/2022 | Thomas Dang | Evaluation of different operating systems |
| 1.2 | 10/11/2022 | Thomas Dang | Added browser evaluation. Completed recommendations. |

**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 client wants to turn their android app game into a web browser version that will run on multiple platforms. The game pulls and image from a library of photos and will begin to draw the photo over 30 seconds. The current team must guess what the puzzle in that time. If they fail, the opposing teams have 15 seconds to make a guess.

## [Design Constraints](#_2et92p0)

The application must be web based.

Each game may have multiple teams and players.

Game and team names must be unique.

Only one instance of the game may exist in memory at a time.

## [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 contains the main() method and is used to launch the application. Program driver uses the SingletonTester class to run tests to ensure only one instance of GameService exists.

Entity is a parent class to Game, Team, and Player. It holds parameters common to all of these classes, specifically id and name, as well as getter methods for these.

GameService contains a list of all games in an instance, and has methods to create unique ids for games, teams, and players. This class is able to create and find games. This class may have 0 or more games associated with it.

Game class represents a single game, and contains a list of teams in that game as well as a method to add teams to the game. A game may have 0 or more teams in its list.

Team has a list of players and the method to add more players, it may have 0 or more players associated.

Player represents a player.

**"The Gaming Room UML diagram. The top of the diagram is labeled as com dot gamingroom. Test boxes are placed in two layers. The first layer has three text boxes and the second layer has four of them. In the first layer, the 'ProgramDriver' textbox points to 'SingletonTester' textbox. The 'ProgramDriver' textbox contains the text 'asterisk main round brackets.' The 'SingletonTester' textbox contains the text 'asterisk testSingleton round brackets.' The arrow between these two text boxes are labeled 'open two angle brackets uses close two angle brackets'. In the second layer, there are 'GameService', 'Game', 'Team', and 'Player' text boxes. The 'GameService' textbox has texts arranged in two layers. The first layer contains games colon List open angle bracket Game close angle bracket, nextGamesId colon long, nextPlayer Id colon long, nextTeamId colon long, and service colon GameService. The second layer contains GameService round brackets, getinstance round brackets colon GameService, addGame open parenthesis name colon String close parenthesis colon Game, getGame open parenthesis id colon long close open parenthesis colon Game, getGame open open parenthesis name colon String close open parenthesis colon Game, getGameCount round brackets colon int, getNextPlayerID round brackets colon long, and getNextTeamId round brackets colon long. The 'GameService' box is connected with the 'Game' textbox with a line labeled 'zero dot dt dot asterisk'.  The 'Game' textbox also contains text in two layers. The first layers contains the text teams colon List open angle bracket Team close angle bracket. The second layer has Game open round bracket id colon long comma name colon String close parenthesis, addTeam open parenthesis name colon String close parenthesis Team, toString round brackets colon String. The 'Game' textbox is connected with the 'Team' textbox with a line labeled 'zero dot dt dot asterisk'. The 'Team' textbox also contains text in two layers. The first layers contains the text players colon List open angle bracket Player close angle bracket. The second layer has Team open parenthesis id colon long comma name colon String close parenthesis, addPlayer open parenthesis name colon String close parenthesis colon Player, and toString round brackets colon String. The 'Team' textbox is connected with the 'Player' textbox with a line labeled 'zero dot dt dot asterisk'. It contains the text Player open parenthesis id colon long comma name colon String close parenthesis and toString round brackets colon String. The 'Game', the 'Team, and the 'Player' boxes point to the 'Entity' textbox in first layer. The 'Entity' textbox contains text in two layers. The first layer has the text id colon long and name colon String. The second layer has Entity round brackets, Entity open parenthesis id colon long comma name colon String close parenthesis, getId round brackets colon long, getName round brackets colon String, toString round brackets colon String.**

## [Evaluation](#_2o15spng8stw)

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Mac has high security and support. There are limited server hosting options available in comparison to linux or windows. MacOS requires using an apple device and can be expensive, with much of the cost going to parts of the machine that are not necessary for a server. | Linux is cost friendly. IT is commonly used in web servers with a significant amount of community support. Prebuilt solutions for web hosting are less common and may require more time adapting for an application, but the result can be more efficient than the prebuilt options available to windows and mac. | Windows is the most common OS. It is more prone to viruses and security vulnerabilities compared to mac or linux. It has plenty of web hosting solutions available as well as a wide range of compatible hardware. If a need to deploy a linux distribution on the system arrises, windows comes with WSL that can help the process. | Mobile is portable, but has poor specs compared to other options for hosting web-based software. |
| **Client Side** | High costs as macOS is tied to apple hardware. Requires a low level of expertise as with uniform hardware many solutions can be readily applied across different projects. Time costs could be expected to be low.  Safari is the default web browser of macOS computers. Other popular browsers are available such as chrome and firefox. | Most linux distributions are free, but can vary greatly from one to another and have any range of specifications. Because of this, it may requires a high level of expertise to setup, and time to develop will depend on that expertise level. Initial costs may be high if an outside expert becomes necessary, but long term can be expected to have the lowest costs.  Linux systems can use most popular web browsers, such as chrome and firefox. | Varying costs. Windows must be bought, but has a wide range of hardware prices. While most windows installations will be similar it will be necessary to plan for a range of system specifications to avoid locking out a high number of clients. Time costs would be expected to be low to moderate.  Edge is the default browser on windows systems. The platform can also support additional browsers like chrome and firefox. | Varying costs. Mobile is primarily dominated by apple and android, and has a wide range in hardware. High level of expertise and high time costs could be expected.  Android devices typically use chrome by default, while apple devices use safari. Most mobile devices from any company are able to install other browsers. |
| **Development Tools** | Mac OS is able to use most languages and development tools available on other systems but may have differing support. Java, javascript, html/css are suggested for the intention of the project. Useful tools include npm, eclipse, postman, atom, and visual studio code. | Most common development tools can be used on linux, but linux may not receive updates and support as quickly as windows or mac. Java, javascript, html/css are suggested for the intention of the project. Useful tools include postman, npm, eclipse, and visual studio code. | Java, javascript, html/css are suggested for the intention of the project, however if a C language is chosen, those languages are better to write on windows machines due to better support from Microsoft without additional costs. Useful tools include eclipse, postman, visual studio, and npm. | Java, swift, kotlin, C#, javascript, html/css are likely to be needed for the project. Cross platform development tools such as appcelerator or Xamarin can help make the mobile development non-platform dependent. |

## 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**: Linux is suggested for a server platform. While there may be some additional training and setup required for a linux based server, costs will overall be reduced due to the many open source options for linux. Additionally, linux is a very common server operating system, so many developers and system admins will already be familiar with a linux server. Lastly linux distributions can easily be setup to provide only what the system needs to function, saving system resources from being used by unnecessary tasks.
2. **Operating Systems Architectures**: Linux is an open-source, kernel based operating system. The major components of a linux system are the kernel, shell, utilities, and the hardware. Linux can be installed on almost any hardware system. The kernel has the low level services of the system. The shell is an interface that communicates with the kernel, it is able to accept commands from the user through a terminal. Utilities consist of applications and programs installed on the system, and includes user installed programs and libraries downloaded.
3. **Storage Management**: Cloud storage can be utilized effectively with linux, and some cloud systems like amazon web services have a linux distribution available for use by the customer by default. Storing the program on a cloud will allow the company to readily access the files remotely. Utilizing cloud storage will also allow the company to create and maintain backups.
4. **Memory Management**: Linux memory management includes implementations of demand paging and virtual memory. It is highly configurable so that memory will only be used when needed.
5. **Distributed Systems and Networks**: The application is browser-based and should, at minimum, be developed with and tested on chrome, other common browsers that should also be tested are safari, edge, and firefox. This will ensure users on any platform will be able to access the game. Utilizing a cloud storage solution can protect the application in the event of an outage
6. **Security**: Most Linux distributions can automatically encrypt their drives as well as an included firewall, which will provide a base layer of security. Cloud servers that host linux, such as AWS, offer expanded security options that are built into the service that can also be leveraged. Additionally, anti-virus programs built for linux are available. For the application code, the code should be reviewed for vulnerabilities such as sql injection or improper use of password hashes and encryption, and to ensure all validation and calculations are occurring serverside, with results passed back to the client.