

Draw it or Lose it

# **CS 230 Project Software Design Template**

Version 1.2

## Table of Contents

[**CS 230 Project Software Design Template** 1](#_Toc115077317)

[**Table of Contents 2**](#_Toc115077318)

[**Document Revision History 2**](#_Toc115077319)

[**Executive Summary 3**](#_Toc115077320)

[**Requirements 3**](#_Toc115077321)

[**Design Constraints 3**](#_Toc115077322)

[**System Architecture View 3**](#_Toc115077323)

[**Domain Model 3**](#_Toc115077324)

[**Evaluation 4**](#_Toc115077325)

[**Recommendations 5**](#_Toc115077326)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.2 | 02/13/23 | Zachary Nicholas | Added 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](#_sbfa50wo7nsh)

The company that has commissioned us would like us to develop a web-based game that is able to run on multiple devices. The game we are to design is called “Draw it or Lose it” which at current is only able to be played on android. The game consists of four rounds each lasting a minute and multiple teams of contestants. When a picture is pulled one team will guess the puzzle until time runs out. If the team cannot guess it the other team will have 15 seconds to guess the puzzle.

## Requirements

The requirements that have been given to us currently are:

* A game will have the ability to have one or more teams involved
* Each team will have multiple players assigned to it
* Game and team names must be unique to allow users to check whether a name is in use when choosing a team name
* Only one instance of the game can exist in memory at any given time. This can be accomplished by creating unique identifiers for each instance of a game, team, or player

## [Design Constraints](#_2et92p0)

The constraints we are given:

* Needs one or more teams
* Each team has multiple people
* Make unique game and team names to allow users to check whether the name is in use or not
* Only one instance of the game at a time
* Must run on multiple platforms

## [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)

The entity creates a relationship between game, team, and player class. With that they will all inherit information from the entity. Using UML we will show this as inheritance, that is so that the individual classes will share common things like their name and their id. Making the entity the parent of the other classes. When we look at the relationships, we see that team will have a player making it a has a type. Game on the other hand has a team, and game service has games. To show these in UML we use the aggregation.

**"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)

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.

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Very useful if you already have Macs on the network and allows easy file transfer and group authentication as compared to windows. Also has a built in GUI that allows for a more friendly user experience.  Generally viewed as a more expensive solution as it needs to be run on apple specific hardware, although if you already have one the buy in price for the server solution is around $20 for the server. | Generally seen as the best “Bang for the buck” solution as it can run on just about any device as long as you are okay formatting a drive or two in order to keep the software running, depending on the flavor you use for a Linux software it can be very inexpensive or you can go for the more expensive options that may add a degree of user friendliness with a GUI.  Generally need someone to help manage certificates as well as group permissions as linux is a lot more open and susceptible to having people getting into places they aren’t meant to if you don’t have someone to manage it | Seen as the same as hosting on a mac server, it will have a nice GUI for user friendliness, it has more software able to take advantage of the resources you give it. Has things like command prompt which allows more knowledgeable users to be able to do things faster.  Problems with this operating systems are requires certain Cpus in order to run, it requires more memory than the other two solutions. | Mobile devices aren’t really the best things to use for hosting as they can come with a host of security problems as well as having little resources as compared to a full server.  Relatively inexpensive but also doesn’t have a lot of options when choosing between different software that can host said website. |
| **Client Side** | Mac doesn’t require a lot of time for a user to be able to connect to the server or interact with their devices, though if its their first time using the operating system it can be a bit jarring as compared to others. Cost is higher than Linux or mobile devices and comparable to windows. | Linux is a hit or miss difficulty wise as Linux offers a bunch of different operating systems that require different knowledge of the platform they are using things like Linux ubuntu and Linux mint require a lot less knowledge of Linux as compared to Arch Linux which is a lot more shell based and needs the user to remember a lot more things to be affective while using it. | Just like Mac, windows doesn’t need a lot of up front work to be done. To use windows effectively as a normal user you just need to know how to use it. Cost is more related to the price of a machine running a similar Mac operating system. | While we may use our mobile devices a lot we are not power users that can look into every nook and cranny of our device and know exactly where things will appear to do this effectively you need a bit of time and a bit of knowledge when it comes to these things. Relatively inexpensive but also requires a lot more expertise to user effectively. |
| **Development Tools** | Mac can code in HTML, CSS, Javascript, python, and swift. Can use things like visual studio code, NetBeans, and homebrew | Linux supports HTML, CSS, javaScript. Some things we can use in Linux is GitHub, visual studio, and eclipse | Windows supports HTML, CSS, and javaScript. IDE’s we can use are Eclipse, visual studio, and GitHub. | Mobile devices can support HTML, CSS, and JavaScript. The IDE’s it can use GitHub, visual studio, and eclipse. |

## 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**: For the application “Draw it or Lose it” I would recommend windows server, mainly because of its ease of use as well as its ability to use multiple types of software. This helps to tip the scale in favor of windows even if the price is a bit harder to swallow. Things that can run along side the game server application are a file server, a web server and even a database server.
2. **Operating Systems Architectures**: Windows will allow a wide range of different applications that will be useful in developing a new application, it has access to things like a detailed user interface as well as access to multiple instant messaging which will prove important during development.
3. **Storage Management**: Windows server will allow us to use things like storage sense and other powerful tools in order to keep the space the application takes up on the hard drive low, with that we can help to keep our overall cost down and focus it on other matters.
4. **Memory Management:** in creating this game we will need to make a library to hold the pictures that will run our game, the more memory we have in order to accomplish this will allow us to store the pictures in the files of the game rather than in a folder under pictures making it more secure.
5. **Distributed Systems and Networks**: In order to accomplish our goal of making an application that will run on any device we need to either seek out an IDE or other test environment that will allow us to test for certain devices on our machine or we need to be able to transfer and test our software on a multitude of different devices and in doing that we will be able to make sure the game runs successfully no matter the device.
6. **Security**: Windows comes in built with its own security software that will help to secure our machines. Given this we can say that windows is the most secure out of the three but we can always add another layer on top of it in order to secure our files and make sure we have no viruses or malware.