

<Draw It or Lose It>

# **CS 230 Project Software Design Template**

Version 1.0

## 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 |
| --- | --- | --- | --- |
| 1.0 | <08/09/2023> | <Ben Douglas> | <I am making a game handler for the multiplayer of the game.> |

**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 problem of the company is that they need a developed web-based game that serves multiplayers. I will be making a multiplayer game. The singleton helps not to duplicate the game. I will be using unique identifiers of the game, teams, and players so that there is only one instance of the game’s memory at a time.>

## Requirements

*<The outline of the design constraints is to keep within a budget. The client’s business and technical requirements is to help with streamlining the development of a web-based game that serves multiple platforms, design a software document, develop the game application, and address the software requirements.>*

## [Design Constraints](#_2et92p0)

<The design constraints for developing the game application in a web-based distributed environment are the budget, the amount of time given to get the job done, and the knowledge that I have in code to get the job done. The implications of the design constraints on the application development are the Java language must work with the web-based game to make the game work.>

## [System Architecture View](#_ilbxbyevv6b6)

The system and subsystem architecture present in the application, including physical components or tiers, are that different players choose different teams to guess at the puzzle to beat the game. A logical topology of the communication and storage aspects are if the players, and teams choose the wrong guess, then the game is over for them, and if the players, and teams choose the right guess, then they beat the game.

## [Domain Model](#_8h2ehzxfam4o)

<The GameService class is the singleton that runs the game. Every game can have zero to a lot of players. The Game, Team, and Player classes are inherited from the parent class of Entity. This allows the classes not to duplicate and restricts the code only to go so far.>

**"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.

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’s characteristics, advantages, and weaknesses for hosting a web-based software are you can run it with other Apple products, but they cost a lot of money. Also, Mac has Apache with CGI that supports multiple programming languages. Another thing is that Mac has VPS to help you keep the costs down on the operating system, and software.> | <Linux’s characteristics, advantages, and weaknesses for hosting a web-based software application are it's an open source, it doesn’t cost money, and a lot of people use it.> | <Windows’s characteristics, advantages, and weaknesses for hosting a web-based software application are its proprietary, costs a lot of money, the makers support it, it’s easy to learn, and it's not as secure.> | <Mobile Devices’ characteristics, advantages, and weaknesses for hosting a web-based software application are it can’t run a server to good, it’s got good battery life, and it’s got good graphics.> |
| **Client Side** | <You must use a MacBook to run Mac, which costs a lot of money. Mac doesn’t have the popularity like Windows does. You must gain knowledge with Mac.> | <Linux doesn’t cost money. It can be used on any of the other three devices. When you start out with Linux, you don’t have to gain a lot of knowledge like Windows, unless you want to gain knowledge about the whole thing, then it can take a while.> | <Windows is the most popularity of the OS. It’s easy to learn. Windows costs a lot of money. Windows has a lot of support.> | <The Mobile Device OS can be easy for the user to understand. It’s not as powerful and easy as a computer OS, like the file explorer.> |
| **Development Tools** | <You would use Objective-C, and/or Swift to code for Mac. Also, you can use Xcode.> | <I would use C to code for Linux. You can also use Java Eclipse for free to code for Linux.> | <I would use C to code for Windows. You can also use C++ to code for Windows.> | <I would use Java to code for Android Mobile Devices. I would use Swift to code for iPhone Mobile Devices.> |

## 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**: <An appropriate operating platform that will allow The Gaming Room to expand Draw It or Lose It to other computing environments is Windows.>
2. **Operating Systems Architectures**: <The details of the Windows operating platform architectures are the User mode, and Kernel mode. The User mode is private to protect the User’s information. The Kernel mode allows developers to create drivers for devices. >
3. **Storage Management**: <An appropriate storage management system to be used with the recommended operating platform, is that in Windows you can delete temporary files to free up space to make it run faster.>
4. **Memory Management**: <The recommended operating platform uses memory management techniques for the Draw It or Lose It software with Windows keeping track of all the memory locations by only allowing one instance of the game to exist in the memory at a time.>
5. **Distributed Systems and Networks**: < I would have the players to be able to log into the server. I would test to make sure that the server and hardware can hold up the players, and games. If they couldn’t hold, then I would either make the server bigger or add more servers.>
6. **Security**: <I would use encryption for the users to prevent unauthorized access to The Gaming Room. Also, I would use antivirus for the users to prevent, detect, and destroy computer viruses if a virus gets on the device.>