

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  2.0  3.0 | 11/11/23  11/26/23  12/10/23 | Sally Keith  Sally Keith  Sally Keith | Executive Summary and Design Constraints and Domain Model  Server Side, Client Side, and Development Tools  Operating Platform, Operating Systems Architectures, Storage Management, Distributed Systems and Networks, Security |

**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 has a game Draw It or Lose It which is available for Android only. They want to develop a web-based game that serves multiple platforms. Using the language Java we should be able to come up with a web-based version of the game that will serve multiple platforms.

## Requirements

*<* Please note: While this section is not being assessed, it will support your outline of the design constraints below. *In your summary, identify each of the client’s business and technical requirements in a clear and concise manner.>*

## [Design Constraints](#_2et92p0)

* The 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.

## [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)

Entity is the parent class of Game, Team, and Player. Game is connected to Game Services. Program Driver and Singleton Tester are independent of the other classes.

**"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** | Terminal that can communicate with the server side, popular in web hosting. Advantages are the terminal and weakness are the high pricing. | Affordable and also has a terminal for server-side communication. Secure and most preferred for server side. Advantage is the low cost and weakness is the lack of programs. | Many programs to choose from. Advantages are the number of programs to choose from and weakness is less secure than the other platforms. | Highly portable and affordable. Advantages are how portable it is and the cost is low. Weakness is also less secure than other platforms. |
| **Client Side** | High expertise  High Cost  Time would be similar to other platforms. | High expertise  Low Cost  Time would be similar to other platforms. | Medium – Low expertise  Medium Cost  Time would be similar to other platforms. | Low expertise  Low cost  Time would be lower as well. |
| **Development Tools** | Xcode  Swift  Apple Developer Tools Languages such as Java and Python. | Vim  Atom  Visual Studio  Geany  Eclipse  NetBeans  Git  Brackets GNU Debugger  Languages such as Java and Python. | Windows API  .NET  WinForms  WPF  Maui  Visual Studio  There are many development tools available for Windows. Languages such as Java and Python. | Android Studio  React  Languages such as Java and Python. |

## 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 recommended operating platform for Draw It or Lose It is Microsoft Azure. This is in part due to the affordable cost and myriad of features that Microsoft Azure has to offer compared to other cloud options.
2. **Operating Systems Architectures**: Microsoft Azure has several architecture types including N-tier, Web-queue-worker, Microservices, Event-driven, big data, and big compute. There is also Gaming using Azure Cosmos DB that can accommodate unpredictable bursts of traffic and delivers low-latency multiplayer experiences. This involves the device connecting to azure traffic manager, where Azure Storage feeds information to Azure Content Delivery Network, and connects to Azure API Apps (the game backend) and Azure Cosmos DB (the game state) which connects to Azure Databricks and Azure Functions and Azure Notification Hub.
3. **Storage Management**: Microsoft Azure includes Azure Blob storage which is optimized for storing massive amounts of data in the cloud. Client libraries are available for different languages including Java and Python.
4. **Memory Management**: Microsoft Azure includes Cache for Redis which is an in-memory-based caching service that provides high performance. It has a store that’s typically used to offload non-transactional work from the database. Memory is significantly faster than disk storage and this in-memory cache enables high performance and scalable architectures. It can handle millions of requests per second at sub-millisecond latency.
5. **Distributed Systems and Networks**: Microsoft Azure has many dependencies such as Java and Python so it can communicate with a wide variety of platforms and devices. The SLA is 99.9 percent so downtime is something that doesn’t have to be worried about.
6. **Security**: Microsoft Azure has security in the forefront and many security features such as the Azure Key Vault and secrets so that no user accounts can access secrets, only the developers. There is also a built-in firewall.