

Gaming Room

# **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 | <09/16/22> | <Emmanuela Filev-Mihalak> | Wrote executive summary, design constraints, and domain model |
| 2.0 | <9/30/22> | <Emmanuela Filev-Mihalak> | Wrote Evaluation |
| 3.0 | 10/14/22 | Emmanuela Filev-Mihalak | Wrote 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 goal of this project is to create a web-based version of the game app Draw It or Lose It. The following software requirements are to be met:

* 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.

We will use our expertise and software to begin the development of Draw It or Lose It for web browsers. Most importantly, the object-oriented programming principles will be applied throughout the project.

## [Design Constraints](#_2et92p0)

Our primary design constraint is the requirement to develop this program in Java. Since we are only developing in Java, we will have to ensure that every requirement is met without the need for other languages to fill in the gaps.

## [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)

<Describe the UML class diagram provided below. Explain how the classes relate to each other. Identify any object-oriented programming principles that are demonstrated in the diagram and how they are used to fulfill the software requirements efficiently.>

This diagram consists of 7 classes: ProgramDriver, SingletonTester, Entity, GameService, Game, Team, and Player. Entity holds common attributes and methods seen in the classes that inherit them. Game, Team, and Player are all inherited from Entity. GameService, Game, Team and Player are all associated, with each class having objects that associate with 0 to many of the other classes’ objects. ProgramDriver uses properties of SingletonTester. Encapsulation is seen in the intentional usage of private attributes. The private constructors Entity() and GameService indicate that this program uses a singleton pattern.

"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.

| **Dev. Reqs.** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Mac has high levels of security. However, the security comes at a high cost, and it is always important to account for the budget of any project. | Linux has access to free software that can be edited by anyone, and therefore is the best choice for cost. However, the freedom it provides means there is less security than on MacOS and Windows platforms. | Windows is held as the standard for operating platforms, so it will have access to the most hosting software, most notably any Microsoft service. | This application was already developed for Android, so we now have to analyze iPhone devices. IOS apps are hosted by TestFlight. Cloud hosting is the most common type of hosting on mobile devices. |
| **Client Side** | Since Apple devices all have so much in common, it is likely that the same expertise needed for iOS programming can be used in developing for Mac. | Due to Linux’s lack of security in comparison to Mac, it is important to consider the time and expertise that will go into securing the application on a Linux client. | While Macs are mostly universal in their specs, Windows computers are sold through many different brands. This means some computers might be able to handle more demanding applications than others, so that is an important consideration. | If submitting an app through TestFlight, one must consider the limit of beta testers (10,000) allowed in each app. |
| **Development Tools** | Swift is the most commonly used program for Mac development. | C++ and Python are commonly used languages in Linux. | Windows supports most mainstream languages, such as Java, C#, Python, etc. | React would be useful for coding a mobile device app, because it makes cross-platform usage possible. |

## 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**:
2. An appropriate operating platform to expand Draw It or Lose It is Windows, because it is a widely used platform for computer games.
3. **Operating Systems Architectures**:
4. Windows has access to essentially every web browser, runs game host applications like Steam smoothly, can host exceptional graphics, and memory can be easily added onto it if a more resource-intensive game is played.
5. **Storage Management**:

An appropriate storage management system for Windows is the combination of an SSD for faster-running games and an HDD for long-term media storage.

1. **Memory Management**:
2. A typical Windows computer uses external RAM to manage memory. If there is an issue with memory management, the computer will show a blue screen with an error labeled MEMORY\_MANAGEMENT.
3. **Distributed Systems and Networks**:
4. Cloud networks are the easiest way to communicate between different platforms. An example is Cloudflare, which hosts many different sites and therefore is reliable with connectivity.
5. **Security**:
6. To protect user information, we must make sure our code is firstly secured. We also create a username and password system for each user, and ensure that login information is protected. This is done by encrypting websites and applications so they cannot be easily hacked into.