

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 | 5/19/23 | Evan Scarborough | Filled blank fields. |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room wants to develop a multi-platform web app for their game Draw It or Lose It. The app will be written in Java using the Eclipse IDE. Server hosting will be required to serve the app to players.

## Requirements

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)

* Will be built in Java, therefore only using Java-compatible packages and IDEs such as Eclipse.
* Must be cross-platform and hosted on the web, code must be written with compatibility in mind
* Must be able to read from a changeable (to keep the game fresh) database of images

## [Domain Model](#_8h2ehzxfam4o)

The GameService class is the foundation of the application, storing a list of the Games which are currently running and methods to facilitate creating unique ids for instances of other objects. Each GameService can have 0 or many Games. Each Game can, in turn, have 0 or many Teams, which can have 0 or many players. The Game, Team, and Player classes inherit from the base Entity class which provides common methods such as for retrieving ids and name information.

**"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.**

## 

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | PROS:  - High compatibility with Mac clients  - Easy to use (MacOS Server includes a GUI)  CONS:  - Scales poorly  - Expensive  - Software & hardware support and availability may be limited | PROS:  - Open source and free  - Secure and reliable  - Runs efficiently, even on less-than-optimal hardware  CONS:  - Requires specialized expertise to operate  - Some software does not run on Linux  - Not all versions are actively supported | PROS:  - High compatibility, support for Windows applications  - Often usable out-of-the-box  CONS:  - Expensive  - Less secure and reliable than Linux | PROS:  - Can be extremely cheap  CONS:  - Hardware is relatively weak and difficult to swap out  - Virtually no support |
| **Client Side** | Supporting Mac OS clients:  - Will require familiarity with Mac OS  - Should include support for finder attributes, resource forks, etc. | Supporting Linux clients:  - Will require familiarity with Linux  - Relatively flexible, shouldn’t require much specialized consideration | Supporting Windows clients:  - Will require familiarity with Windows (not likely to be an issue)  - Well-supported, not likely to require much specialized consideration | Supporting mobile clients:  - Will require familiarity with the mobile OS  - Will require a separate GUI due to differences in screen size & aspect ratio |
| **Develoent Tools** | - Database organization with SQLPro or Oracle Database (Java-based)  - API testing with cURL (included with OS)  - Many server options  - Support for Eclipse, IntelliJ IDEA | - Database organization with Oracle Database (Java-based)  - API testing with cURL (may be included with OS)  - Many server options  - Support for Eclipse (via tarball installer file) | - Database organization with Oracle Database (Java-based)  - API testing with cURL (included with Windows 10)  - Many server options  - Support for Eclipse, Visual Studio .NET, etc. | Development is not done using mobile devices but the following applications are used to develop for them:  - Eclipse, Visual Studio, many others  - Swiftic, React Native, weex, others  Due to smaller, less powerful hardware, demand for client resources should be light. |

## 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**: I recommend using Linux as the server OS. Linux requires more expertise than Windows or Mac, but it also allows for a greater degree of flexibility which will be advantageous when maximizing compatibility is a priority. Linux servers are also cheap, secure, and highly available.
2. **Operating Systems Architectures**:

The Linux architecture consists of:

* + Hardware: the physical components
  + Kernel: Allocates resources to each process running on the OS and ensures appropriate abstraction, contains all functionality in a single executable. Configurable.
  + Shell: Command-line interface, allows user to interact with kernel processes while providing abstraction.
  + Application: The programs run using the Linux OS.

1. **Storage Management**: Linux includes built-in storage management tools; Oracle Database will also be helpful for management of database resources.
2. **Memory Management**: Linux uses a hierarchical system of memory management that organizes data into a tree structure. Partitions can be created and inserted into this tree. The built-in *df* command displays information including the remaining memory available.
3. **Distributed Systems and Networks**: First, the client will make a request of the Linux server using the browser on their device. The server should receive information about the type of device making the request and use this to determine how to respond. The information the server provides will be displayed client-side in a format appropriate for the client’s platform. Caching will be used to reduce server load and preserve some functionality in case of outages, etc. The server serves as the point of contact for all clients; thus, as long as the client can communicate with the server, it can also communicate with any other client through it.
4. **Security**: The central role of the server in protecting user information and ensuring the availability of Draw It or Lose It make it a priority for security considerations. The physical server should be kept in a private room with access restricted only to essential personnel. A username-password system should be used to authenticate users who attempt to access the game, and passwords should be encrypted and stored with the server, only communicated over the internet using secure protocols such as HTTPS or SSH. Lastly, tests should be performed regularly to scan for potential security vulnerabilities within the network, server, and game application itself.