Project One – Draw It or Lose It

Joshua Asari

CS 230-T3636 Operating Platforms

Dr. Lyon

February 20, 2020

## 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 | <1/30/22> | <Joshua Asari> | <Initial Writeup for the Project Software Design> |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room’s mission is to develop a web-based game that serves multiple platforms based on their current game, Draw It or Lose it, which is only currently available on the Android app.

## [Design Constraints](#_2et92p0)

Because of the requirements given from the client only relates to software requirements, there are only technical constraints within the project.

The first technical constraint is platform constraint. This project must be developed to satisfy the web-based platform.

The next technical constraint is to choose a certain programming language in order to develop the project. Within this project, the main goal is to convert an Android app to a web-based app. Because it is a web-based app, a cross-platforming framework must be chosen for ease of transitioning. These frameworks for example include React Native / Flutter / Node.js / etc.

## [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)

Starting with the base class Entity. It has two properties of private long id and private String name. Its methods include a private default constructor and a public default constructor with two parameters. It has three more methods, public getId(long), public getName(String), public toString(String).

These properties and methods are all used by the classes Game, Team, and Player as they extend Entity. Because of this, the classes Game, Team and Player inherit the base class Entity. This means any Game, Team or Player object all inherit the two properties and methods included in the base class.

Looking at the UML Diagram, there are also a relationship between GameService, Game, Team and Player. GameService has a cardinality of zero or more to Game. Game has a cardinality of zero or more to Team. Team has a cardinality of zero or more to Player.

This means that GameService can create as many games that is necessary. Each Game that is created by GameService can also create as many Team within its own Game. Lastly, each Team within a Game can also have as many Players within it as necessary.

Now the GameService class has 5 properties, and 8 methods. The Game class has 1 property and 3 methods. The Team class has 1 property and 3 methods, and lastly the Player class has 0 properties and 2 methods.

The GameService class has a property of GameService service alongside a method called public GameService getInstance(), in order to make sure there is only one instance of GameService running.

Lastly, there are two classes called ProgramDriver and SingletonTester. ProgramDriver is where main() resides, and SingletonTester has one method, public testSingleton() in order to test if the program is a singleton. Because of this, the ProgramDriver uses the SingletonTester class in order to test the Program’s singleton requirement.

**"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)

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Mac has a simple and powerful user interface. Its terminal is also flexible to design the server. | Linux is difficult to understand and will need time and patience to become familiar with it. Linux contains many low-level tools which gives Linux its versatility, power, security and speed. | Windows has much more supported software compared to Mac or Linux. Its user interface is also friendly. | There are many different Mobile Devices that may change this specification. |
| **Client Side** | Mac is an expensive option for most users. Since its user interface is friendly, the expertise needed is not high. Lastly, time is also low as well. | Linux requires the most amount of expertise and time to use. Although, when compared to Mac, it has lower cost. | Windows has minimum expertise and time required as most people are already familiar with Windows. It’s cost however is between Linux and Mac. | Mobile Devices has a higher expertise and time requirement as applications on a web-based platform would perform better on a Mac, Linux or Windows OS. |
| **Development Tools** | Programming Languages for web-based application include Python, Java, PHP, Javascript, HTML, CSS. Mac IDEs include Visual Studio Code, Atom by Github, Sublime Text 3, PyCharm, Eclipse etc... | Programming Languages for web-based application include Python, Java, PHP, Javascript, HTML, CSS. Linux IDEs include Atom, PhpStorm, NetBeans, Webstorm etc... | Programming Languages for web-based application include Python, Java, PHP, Javascript, HTML, CSS. Windows IDEs include Microsoft Visual Studio, Atom, NetBeans, PyCharm, Eclipse etc... | Programming Languages for mobile devices include Swift and Object-C for iOS and Java, Kotlin, C#, Lua, Basic for android. Mobile IDEs include Visual Studio, PyCharm, Atom, Eclipse etc... |

## 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 this project would be the Windows Operating System. This is because, for the client side, it has the minimum amount of expertise and time required to support. Its cost is also lower than using a Mac OS. Windows also has a myriad of supported software when trying to decide which IDE to use. For converting an android application to a web-based application, Windows would be very user friendly.
2. **Operating Systems Architectures**: The Windows OS Architecture is a modular structure, composed of several simple modules. These modules include the Hardware Abstraction layer, Kernel/microkernel, Executive Services, Environment Subsystem and Integral subsystem. These modules make up the two main components of user mode and kernel mode. Kernel mode has unrestricted access to the system memory and external devices, while user mode is made of subsystems capable of passing I/O requests to the appropriate kernel mode.
3. **Storage Management**: The client must maintain a server room for its storage management. If the client does not want to do this, a cloud-based server may be chosen instead. There are many cloud storage service providers out there such as AWS, Microsoft Azure and Google. Between the three, AWS is presumed to be the least expensive option available. Looking over their systems, the Amazon Glacier allows the client to query data in place and retrieve only the subset of data you need from within an archive. This can be advantageous for the client and it only costs All Storage / month $0.00099 per GB. Compared to google cloud storage, their cheapest option is $0.004 GB / month. Lastly, Microsoft Azure’s cheapest option is $0.002/GB per month. If going the cloud storage route, I would recommend AWS Glacier.
4. **Memory Management**: Windows uses a Virtual memory system where processes use virtual addresses that are translated into physical addresses in blocks called pages. Windows uses Paged memory management. This system is of course built into Windows, and the application must be developed in a way where the database of images will be stored in this system in order to be retrieved later when needed.
5. **Distributed Systems and Networks**: Windows OS can set up a cross-platform network in order to communicate between various platforms. This cross-platform network will be able to communicate with the server room / cloud server as well as the different OS that the end users will be using. This is called the client-server model, and it is a distributed application structure that partitions tasks or workloads between the providers of a resource.
6. **Security**: Windows is the Operating System that is the most susceptible to having poor security. Extra development time must be invested in order to encrypt data and protect user information.

**Reference**

*15 best web development ide in 2022[updated]*. Hackr.io. (n.d.). Retrieved January 31, 2022, from <https://hackr.io/blog/web-development-ide>

Chapel, J. (2019, June 4). *Cloud storage cost comparison: AWS vs. Azure vs. google*. ParkMyCloud. Retrieved February 20, 2022, from <https://www.parkmycloud.com/blog/cloud-storage-cost-comparison/>

*Client-server model*. GeeksforGeeks. (2019, November 15). Retrieved February 20, 2022, from <https://www.geeksforgeeks.org/client-server-model/>

Stacksol Follow. (n.d.). *Windows Architecture explained by Stacksol*. SlideShare. Retrieved January 31, 2022, from <https://www.slideshare.net/Stacksol/windows-architecture-explained-by-stacksol#:~:text=The%20architecture%20comprises%20HAL%2C%20driver,functions%20of%20the%20operating%20system>.

UML class diagram tutorial. (n.d.). Retrieved January 31, 2022, from <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/#:~:text=The%20UML%20Class%20diagram%20is,classes%2C&text=and%20the%20relationships%20among%20objects>.