

<Name-of-Software-Application>

# **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 | 3/19/2022 | Nick Burris | Adding summary |

**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, and they want to expand their game from android only to a multi-platform game. The game itself is a team-based game, where each player guesses a puzzle created from stock images. With only one guess per team, and a 15 second time limit after the 30 seconds of images being rendered.

## [Design Constraints](#_2et92p0)

Many of the constraints are based on how many players can be in one game, amount of teams in one game. How many rounds the game goes through and or how many points until the team wins.

## [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)

Through the UML Diagram we can see the Game Service starts by getting the instances of the games, players and teams. Then we see the Game instance set the teams in the games, and the team class set the players on each team and the player class initiating each player as a unique instance. While entity creates the bridge between plater team and game.

**"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 has been pulling away from the server side of applications since 2009, Mac OS X also does no work with J2EE development, | Linux, has the least amount of security by design, but because it has less code than windows and less users makes it more secure. | Windows has lots of developmental tools, but doesn’t have very good security, security measures would need to be added when developing and would cost more. | Mobile devices have much less processing power meaning they are slower and need a lighter load to run, but are secure and have a larger audience base. |
| **Client Side** | For mac developers there is a 100$ fee per year and you have to use a mac and you need to pay for most software used to develop making mac the most expensive option | Linux is the most used for web hosting, and without having to pay for any development tools or licenses It is the cheapest option. It also is highly customizable based on the needs of the project. | Windows can use a use a plethora of different platforms but the trade off is compatibility which can still be done, it takes more time and energy compared to Linux for and web based application | For mobile devices Mac is the most expensive to develop, but has the largest audience. While android is easily done with windows and Linux based systems. |
| **Development Tools** | Mac uses XCode, Which uses the swift language. Which gives the flexibility to write in several different languages. | Linux for development is one of the best you have everything you need for free with quick installs. | Windows has several different tools available the main issue is compatibility and support for the tools on windows. | If its IOS, you need XCode and a mac to develop, which is expensive. While a Android just needs a Windows/Linux based IDE Which is much cheaper |

## 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 would recommend Linux, It has the most developmental tools and is the most commonly used for web based applications and can easily be sent to windows if need be. Also because of Linux’s open source, people are scouring the code and constantly fixing code to make the operating system more secure. Using Linux is also the cheapest option with the operating itself being free and the developmental tools also being free. Also Linux has a multiple user tool meaning the sharing of resources for a team, and multiple people working on the same project at the same time is very beneficial to the developers.
2. **Operating Systems Architectures**: Linux has a very similar architecture to windows, having a kernel, shell and application. The Kernel is the core section of the operating system, which is responsible for all the major actions of the OS. Because of the major actions the kernel has the ability to do many of its actions are abstract and hide details to be secure. This is where the shell is used as a interface between the user and the kernel to run applications and functions. The shell takes the commands of the user and runs the functions of the kernel.
3. **Storage Management**: When it comes to storage management AWS, would be one of the better options. AWS offers the most competitive price and have been out performing google. So the package I would suggest is Amazon EBS which boasts low latency and consistent performance for .10 cents per gigabytes of storage per month.
4. **Memory Management**: I recommend using a dynamic based loading system, Which means data is loaded only when its called instead of having everything loaded when the program starts up. While this is happening memory allocation will be important to stop the fragmentation of data.
5. **Distributed Systems and Networks**: The model that should be used for this is the client server model. Where all the data needed for application is stored in one place, which creates a easy protection of data and users and authentication. Now for the communication between server and client. The client starts by sending a request to the server and the server sends back a response. The client also takes requests and organizes them in a prioritized scheduling system.
6. **Security**: Using a client based service most private information is stored within the clients servers means that all stored information is secure. The main threat is the communication between client and user, Which means the client will need to ask the user for there password and user name before they can use the client.