

Draw It or Lost It

# **CS 230 Project Software Design Template**

Version 1.1

## 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 |
| --- | --- | --- | --- |
| 1.2 | 10/14/2022 | Sulochana Pradhan | Revised recommendations. |
| 1.1 | 10/02/2022 | Sulochana Pradhan | Updated operating system evaluation. |
| 1.0 | 09/17/2022 | Sulochana Pradhan | Initial Document. |

**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)

Client: The Gaming Room

The Gaming Room wants to develop a web-based game, Draw It Lose It, which serves multiple platforms. Currently, there is only an android application. The key idea of the game is to allow multiple players to play four rounds, each lasting a minute. In each round, a picture is pulled from the library and one team guesses the puzzle till the time expires. If the team doesn’t guess the puzzle, the remaining teams have an opportunity to solve the puzzle within 15 seconds.

## [Design Constraints](#_2et92p0)

1. The game supports one or more teams.
2. Each team can have multiple players.
3. The game name and team name are unique.
4. Each player on the team has a unique name.
5. The game needs to support all modern browsers.
6. Only one instance of the game can exist in memory at any given time.

Each of these constraints will be applied while developing the game. As there is already an android version of the game available, the development team can reuse or reverse engineer the business logic of the android application. The game needs to be in HTML5 and will run in all modern browsers. The development team can develop responsive HTML5 web applications so that it is easy to create mobile and tablet applications for different platforms if needed.

## [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)

The Entity is the base class from which other concrete classes: Game, Team, and Player are derived. All these derived classes share states and behaviors from the base Game class. The private parameter-less constructor of Entity makes sure all instances of Entity have id and name. GameService class generates a unique identifier for Game, Team, and Player objects. The same service has 0 or more Game instances, the Game class has 0 or more Team instances, and the Team class has 0 or more Player instances. All these relationships are association relationships. GameService and Game, Game and Team, and Team and Player each have an association relationship defined by a line.

**"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.

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | -Mac supports server-based deployment.  -Great user interface and responsive.  -It comes with an Apache server.  -Moreover, it supports different other web server installations.  -It supports vertical scaling to support an increasing number of users.  -It is built for the general user and for development rather than hosting. It needs its hardware and is expensive.  - It is preloaded with many applications which are not needed for hosting.  -It has excellent customer service support. | -Linux supports the server-based deployment.  -It is popular and preferred for web hosting.  -Most of the deployment process is done via its powerful bash terminal.  -It supports both horizontal and vertical scale and can supports huge user load.  -It is free and open source with huge open-source community to back it up.  -There are various versions and distributions. For our purpose we can use bare minimum distribution such as Alpine Linux.  -There is customer service and with some cost, premium support is also available. | -Windows supports server-based deployment.  -It provides different versions and Windows Server distribution is popular for hosting.  -It supports both horizontal and vertical scaling to handle user load.  -It comes with the popular directory service: Active Directory for handling users.  -Similar to Linux, the server version comes with minimum applications.  -It has a licensing fee and the operating system is proprietary.  -It provides excellent customer service. | -Mobile supports the limited server-based deployment.  -Not enough web hosting applications available.  -It is difficult to scale mobile devices to handle user load.  -It is expensive and the least preferred for web server deployment.  -Even though there is good customer service, the purpose of the device is not for hosting application. |
| **Client Side** | -Mac supports all modern HTML5 browsers. These browsers support testing responsive behavior.  -It comes with XCode for iOS emulator. Android studio can be installed to emulate the android application.  -It is popular among web developers as application installation is straightforward.  -It has a huge number of development tools available.  -Cost-wise it is expensive as it needs its hardware.  -The operating system itself is very responsive and provides a smooth experience to the developers. | -Some Linux distributions support the GUI. Different modern HTML5-based browsers can be installed and used for testing responsive behavior.  -Software installation is not as easy as windows and mac.  -There are fewer applications than mac and windows.  -The android studio can be used to emulate the android app.  -Paid emulators can be used to emulate the iOS app. It adds up to the cost. | -Windows supports all modern HTML5 browsers. These browsers support testing responsive behavior.  -It has a huge number of development tools.  -It is popular among web developers and application installation is straightforward.  -It is cheaper than mac but more expensive than Linux.  -The android studio can be used to emulate the android app.  -Paid emulators can be used to emulate the iOS app. It adds up to the cost. | -Even though, it supports modern HTML5 browsers, it is not ideal for testing responsiveness.  -Difficult to develop an application.  -Not enough applications are available for developers.  -Takes a lot of time to write code and test.  -Best for testing its own version of apps. |

| **Development Tools** | -Free Visual Studio Code can be used as an IDE. Other free alternatives are Eclipse, IntelliJ, atom etc.  -GitHub private repository can be used for version control.  -To save cost, GitHub issue can be used as project management tool.  -Free slack service can be used for communication among the team members.  -Supports development of application based on different programming languages: C, C++, PHP, Java, NodeJS, Python, .NET core etc.  -Supports major database servers: MySQL, MSSQL, Oracle, MongoDb etc.  -Supports client-side application development using HTML5, JavaScript, CSS based application using modern JavaScript frameworks: VueJS, Angular, React etc. | - Free Visual Studio Code can be used as an IDE. Other free alternatives are Eclipse, IntelliJ, Atom, etc.  -GitHub private repository can be used for version control.  -To save cost, GitHub issues can be used as a project management tool.  -Free slack service can be used for communication among the team members.  -Supports the development of applications based on different programming languages: C, C++, PHP, Java, NodeJS, Python, .NET core, etc.  -Supports major database servers: MySQL, MSSQL, Oracle, MongoDB, etc.  - Supports client-side application development using HTML5, JavaScript, and CSS-based applications using modern JavaScript frameworks: VueJS, Angular, React, etc. | - Free Visual Studio Code can be used as an IDE. Other free alternatives are Eclipse, IntelliJ, Atom, etc.  -GitHub private repository can be used for version control.  -To save cost, GitHub issues can be used as a project management tool.  -Free slack service can be used for communication among the team members.  -Supports the development of applications based on different programming languages: C, C++, PHP, Java, NodeJS, Python, .NET core, etc.  -Supports major database servers.: MySQL, MSSQL, Oracle, MongoDB, etc.  - Supports client-side application development using HTML5, JavaScript, and CSS-based applications using modern JavaScript frameworks: VueJS, Angular, React, etc. | -Different development teams are required for iOS and Android application development if we choose XCode and Android application development.  -To save development costs and utilize the same developers we can use HTML5-based mobile applications based on the same client-side code base. We can use Angular, Ionic, React Native, etc.  -we can also use a separate development team but a common framework such as Flutter to use the same codebase for iOS and Android.  -To publish application on app store, we need apple developer account that costs certain amount each year.  -Needs flat onetime fee to publish application in the Android play store. |
| --- | --- | --- | --- | --- |

## 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 like to recommend 64 bits Windows for hosting The Gaming Room. The Windows server is ideal for hosting applications and is not preloaded with different unnecessary applications. It is simple to use with a great user interface. Moreover, it has a built-in internet information system (ISS) to host our application. The modern version of the windows server provides a Linux subsystem which can be also used if we want to harness the feature of Linux for our gaming server.
2. **Operating Systems Architectures**: The windows operating system has kernel mode which is responsible for scheduling, thread prioritization, memory management, and interaction with hardware. The kernel has full access to the hardware and system resources. Another mode: user mode is responsible for passing input/output to the appropriate kernel mode drivers. The windows operating system has a powerful graphical user interface and comes with 32 bits and 64 bits versions. Any performance tuning and optimization is handled by the kernel which is the central part of the operating system. This operating system contains a process that holds threads. These threads execute the instructions of the program.
3. **Storage Management**: We can utilize cloud services such as AWS or Azure for our storage. In the case of AWS, we can utilize s3 to store our image. If we need faster access to our assets, we can use the amazon elastic block store. We can use a containerized solution such as an elastic container service and repository service to store the docker image and host the service. The key advantage of using cloud services is we do not need to maintain servers and we can easily increase or decrease storage capacity as needed.
4. **Memory Management**: As we are using 64 bits Windows operating system, the automatic configuration allocates up to 16 TB of accessible memory space to the kernel and user programs. Each process has 8 TB address space. The process contains multiple threads which have access to the visible address of the process. To protects the memory space of the different process, threads from one process cannot access the memory space of the other processes. The operating system itself is responsible for managing the memory for individual process. However, we can change it as we want and give more priority to our application so that it can consume more memory. Ideally, it is not recommended and if operating system is running out of memory, we can easily do horizontal or vertical scaling to increase the memory.
5. **Distributed Systems and Networks**: To make sure the application on different platforms can talk to each other, we can define the contract which serves as the way to make the communication between the server and client. I would recommend a REST endpoint and use it to communicate with each other via the endpoint and predefined contract. To distribute the user loads, we could use the load balancing in the Windows Server. We can host our game in multiple locations so that, even in case of a power outage in one area, the game hosted in another area still serves the requests. Ideally, we could use cloud platforms such as Amazon Web Service, and Azure and use different services such as availability zones, load balancing, and autoscaling to make our game highly available. To provide faster access to our static assets such as high-resolution images, CSS, html, and JavaScript files, we can use the content delivery network (CDN). Most of the cloud providers already have that.
6. **Security**: AWS has a Cognito service that can be used to manage user information including login. This way, we do not need to build the authentication system from the scratch. The user will be able to log in to our system using Cognito login which supports millions of users. We can use SSL communication between the client and server. User information and communication can be encrypted and stored in a virtual private network. Only our application can communicate with the user database. Moreover, every client request needs to be protected by an OAuth token which the client can receive during login. We can use an auto-rotating key/secret combination to identify if the client is authentic or not. To protect from DDoS and bots, we can use the AWS web application firewall (WAF).

**References:**

* Operating Systems Architecture. (n.d.). Olympic College. Retrieved October 14, 2022, from <http://cis2.oc.ctc.edu/oc_apps/Westlund/xbook/xbook.php?unit=04&proc=page&numb=9>