

Draw It or Lose It

# **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 | 03/15/22 | Denis Dzenyuy | Addressing software requirements |

**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 is seeking to develop a web-based gaming application called ‘Draw It or Lose It’ that will run on multiple platforms based on their current game. Currently the game is only available on android app. The concept of a game in ‘Draw It or Lose It’ consists of four rounds of play lasting one minute each. In each game which will have the ability to have one or more teams consisting multiple players involved, drawings are rendered at a steady rate and are fully complete at the 30-second mark extendable by another 15-seconds (time limit) for other teams if a team fails to meet guessing conditions. A game will consist of four rounds of play lasting one minute each. The concept of the gaming application involves images being rendered from a large library of stock drawings as clues.

Considering the fact that the game will require the ability of hosting multiple teams with unique names, and several players, and images in the game will be rendered at a steady rate and fully complete at the 30-second mark. This application will have to be capable of operating at a high capacity and be able to handle heavy web traffic (data). It also will have to have the right security system/protocol, as well as it needs to be light enough not to take up too much compute capacity. The platform (Content Management System) has to be user friendly, works across all devices/channels, and flexible. We may also need to consider scalable web-based game hosting infrastructure(s).

## [Design Constraints](#_2et92p0)

* Needs to have the ability for one or more teams involved per game
* Each team can have multiple players per game
* Game and Team names must be unique to allow users to check whether the name is in use or free
* Only one instance of the game can exist at any time.
* Must run on multiple platforms

These are the software requirements that will need to be followed in the process of writing code and designing software for the game. While this is only the game aspect, we may run into browser compatibility, as well as UI related constraints due to differences in platforms/Operating systems or development environments. There is also the issue of device compatibility since the application will have to run on different types of web-based devices (desktop, laptop, and mobile) running Windows, Linux, Apple etc. We will also need to dedicate resources towards determining the right coding approach such as code re-write in swift (for Apple devices), using of existing code, or implementing a cross-platform language like python that will easily inherit other languages as well as run on all systems.

## [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)

An entity creates a relationship between Game, Team, and Player class. Hence, they all inherit or get information from Entity. In the below UML this is illustrated with inheritance. Each class shares a common reference such as “name” and “id”. Making Entity a superclass. When we look at their relationship, we see Team and Player is a “has a” type. While Game has a Team and GameService has Games. When we use UML, we call it aggregation (HAS-A). When a user “has a” this means it is an instance of one class and has a reference to an instance to another class. As depicted in this diagram, GameService has a reference of Games, Games a reference of Tea, and Team a reference of Player.

**"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)

The objective is to create a web-based application thus a server-style configuration (server-side scripts) for hosting the website that is allowed to scale up to thousands of players (using client-side scripts). This in other words means that the application or game is to be hosted in a distributed environment, and network intensive (game running on the web application servers) setup where input is collected via client applications, processed at servers and output is rendered on the user interface (UI).

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Mac has flexible terminal commands to configure the server, access, or make changes.  - **Characteristics:** Continuity and handoff, best OS/OP ecosystem integration, It is relatively popular in web hosting  **Advantages:**  - It is upgradeable, - It has various options for different web hosting requirements  **Disadvantages:**  - It is highly proprietary and less preferred for web hosting services | Consistent growth and popularity. **Characteristics:**  - Cost effectiveness.  - It is secure, and provides privacy.  **Advantages:**  - Security flaw monitoring is real time meaning issues are caught before they become an issue.  - It is the most preferred choice for web hosting services.  **Disadvantages:**  - Even though it is open source, It is more difficult to find applications to support the web hosting required needs. | High software availability compared to other OS.  **Characteristics:**  - It is most compatible with and dominant to the other platforms.  **Advantages:**  - High resource requirements.  - Less loading time.  - Lowest stability rate.  - Frequent reboots.  - Easy virus susceptibility. | Strong support community, better coding capabilities.  **Advantages:**  - Cost effective.  - Light on the compute capacity side.  - High and better portability.  - More popular.  - Has wider reach. **Weaknesses:**  - Sever can hardly be tracked in a single location.  - Software is highly selective to various smart mobile devices.  - Security is relatively weak. And,  - Software design specifications are better in other platforms. |
| **Client Side** | - User friendly.  - Widely preferred.  - Moderate expertise and time required to apply.  - Relatively expensive.  - Stability is high and compatibility or cross functionality with other OS/OP’s is limited.  - Web browser, platforms and mobile devices are proprietary hence require minimal expertise. | - Highest level of expertise and time required.  - Minimum cost.  - The amount of effort required in the application development process to ensure compatibility with all web browser platforms and mobile devices is high. | - Due to high adoptability and support services, the expertise and time required is minimal.  - Cost is similar to mac.  - The amount of effort required in the application development process to ensure compatibility with all web browser platforms and mobile devices is low. | - High availability due to “mobile” nature.  - Provides flexibility to clients as well as developers to see updates in any place.  - The amount of effort required in the application development process to ensure compatibility with all web browser platforms is slightly more difficult to implement than other devices. |
| **Development Tools** | Mac natively uses Xcode written in Swift programming language, C#, Objective C, and/or C/C++.  **IDEs:**  - Xcode 8 IDE.  - Appcode.  - Atom.  - SublimeText 3.  - CodeRunner 2. **Code editing:** Notepad++.  **SDK:**  - Cocoa Touch.  - Frmeworks.  - Beta.  **Testing:**  iOS Unit test or XCTest framework.  **Cloud Testing:** Xbox cloud testing for iOS.  **Deployment:**  - PhoneGap.  - CocoonJS.  - AWS CodeDeploy. - GameClosure. These will help Mac web-based games running on languages consisting of but not limited to HTML/CSS/JavaScript while supporting libraries to support the frontend and general-purpose languages. These can be Java, Python, PHP, and Ruby. | C is commonly the language that Linux is written in. There are wide varieties of game engines for JavaScript as well as C#.  **IDEs:**  - LinuxGSM.  - GameMaker.  - Linux. **Deployment:**  **-** GitLab CI/CD.  - Jenkins.  - AWS.  - CodeDeploy.  - TeamCity. Or,  - ManageEngine.  - Desk Central. **Code editing:**  - Visual studio.  - Code.  - Eclipse, along with notepad++. **Compiler:**  - Eclipse. Along with many more languages and tools.  These will help Linux web-based games running on languages consisting of but not limited to HTML/CSS/JavaScript while supporting libraries to support the frontend and general-purpose languages. These can be Java, Python, PHP, and Ruby. | - Visual Studio Code though easier to use than Linux, it can run the same languages as Linux **IDEs**:  - So visual studio. - Project Rider.  - Visual Studio.  - Atom.  - Mono Develop.  - Develop 4. And, - Eclipse to name a few of the tools windows uses. **Deployment:**  - AWS CodeDeploy.  - Octopus Deploy. - TeamCity.  **Code Editing:**  - Notepad++.  These will help Windows web-based games running on languages that consist of but not limited to HTML/CSS/JavaScript while supporting libraries to support the frontend and general-purpose languages. These can be Java, Python, PHP, and Ruby. | - Some of the most popular mobile game engines and development platforms are;  - Unity.  - Construct 3.  - Buildbox.  - Solar2D(ex Corona SDK) Spritekit – 2D sprite-based games, and,  - Fusion 2.5.  These can help create countless apps using android and swift. Both languages and software can be run on most platforms or machines.  These will help Mobile device/ games running on languages consist of but not limited to HTML/CSS/JavaScript while supporting libraries to support the frontend and general-purpose languages. These can be Java, Python, PHP, and Ruby. |

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**:

Considering Windows wide adoptability and use, I would recommend The Gaming Room to start on windows devices as it also has more software available along with minimum expertise and cost to fully develop and deploy the web-based version of the application and eventually expand to other environments. There is also the added advantage of not running into a shortage of IDE’s to work with.

**Operating Systems Architectures**:

Windows architecture is highly modular, and consists of two main layers: a user mode and kernel mode. Programs and subsystems in user mode are limited in regards to what system resources users have access to, while kernel mode has unrestricted access to the system memory and external devices. Windows also mixes native games, support for retro titles, and it has the ability to setup any contemporary game to maximum resolution settings, and better software optimization, and provides services used by all Windows-based applications that enable applications to show a Graphical User Interface (GUI) while accessing system resources and much more. These applications also refer to Graphics and Multimedia, messaging, and web services.

**Storage Management**:

The latest version of Windows (Windows 11) is fully compatible with one of the most popular scalable cloud storage services (Microsoft Azure Cloud Storage Solutions and Services) which happens to be part of Microsoft with is the Mother Company of both Windows and Azure line of products and services. Windows also comes with a management feature called storage sense. This allows you to scrutinize and manage files locally on your hard drive, along with space capacity management. Other features include being able to choose to save locations for apps making them easier to find. In addition, there is a built-in storage system, which allows for easy file creation and placement for large projects, which provides protection against lost or accidental deletion.

**Memory Management**:

Windows has both physical and virtual memory. Memory is managed in pages, with processes demanding it as necessary, so for the Draw It or Lose It software, a database or library with significant processing power. Windows divides on the game’s architecture the total addressable memory into user space and system space. The memory allocation allows for easy storage of pictures outside of the default picture folder. This also allows during development for the whole project to be kept together in a more secure area on your system/computer. This includes when working with your IDE and opening files from it to create the game. Per Windows recommendations, each process on 32-bit Microsoft Windows has its own virtual address space that enables addressing up to 4 gigabytes of memory. Each process on 64-bit Windows has a virtual address space of 8 terabytes. All threads of a process can access its virtual address space. However, threads cannot access memory that belongs to another process, which protects a process from being corrupted by another process.

1. **Distributed Systems and Networks**:

Considering the fact that each operating system or operating platform is different Draw It or Lose It may have to be built as a Hybrid Cross-Platform App where it is built inside a native container using each platform’s browser built-in component, allowing it to run on each platform as if it is a native app on that platform. This can be accomplished by using HTLM5, JavaScript and CSS. As a Hybrid cross-platform app, Draw It or Lose It also solves the problem web apps face in regards to internet connectivity, where it does not require internet connectivity since it is running as a native shell. Hybrid apps can also be published on mobile app stores where users can easily find them, hence increasing discoverability and consequently the number of users. This will help with dependencies. To prevent outages or connectivity, the company will need to make sure their servers have enough compute capacity to support large gaming power/player volumes along with backup power for power outages.

1. **Security**:

Windows comes with built-in security protection software. However to secure user data and information, emphasis and serious consideration on data protection will be laid when developing the game’s software as this may lead also to significant competitive advantages. Data controllers are obliged under the GDPR to minimize data collection and processing, using, for instance, anonymization or pseudonymization of personal data where feasible. Data encryption, implement features that prevent exposure of personal information, limit data use, implement elaborate password policies, hashing, two factor authentication, ensure web security by protecting different vulnerabilities. Windows also comes pre-equipped with protection, which allows the system to scan for malware, viruses, and security threats. This all happens in real-time, and because threats change the system updates automatically to keep the system and user information safe