

Draw it or Lost It

# **CS 230 Project Two: Software Design**

Version 3.0

## 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 |
| --- | --- | --- | --- |
| 3.0 | 04/16/2023 | Tyler Walker | The final section (“Recommendations”) has been completed, summarizing the recommended development platform and its characteristics in terms of storage management, memory management, security, architecture, and distributed systems. |

## [Executive Summary](#_sbfa50wo7nsh)

“The Gaming Room” game development company needs Creative Technology Solutions (CTS) to facilitate the expansion and conversion of their current game “Draw it or Lose it” from an Android-only based application, into a web based cross-platform system that users can access from any device with a web browser. The solution to the problem presented is as follows: This expansion will require an overhaul of the current system starting with implementing the logic of the current application into a language for web-based development. Object oriented programming principles will be utilized to create classes for teams, players, and game sessions, that will control the flow of the game and ensure the game functions correctly, according to the technical and functional requirements provided by “The Gaming Room”. Object Oriented Programming will also ensure the environment is easy to understand, operate, and maintain when it is handed over to “The Gaming Room”. The above-mentioned requirements are included in the next “Requirements” section, as well as the business requirements.

## Requirements

**Business requirements**

* The game serves multiple platforms.
* CTS facilitates the development of the game.
* The environment developed by CTS will fulfill all the same technical and functionality requirements as the existing Android Application. This will be beneficial from a user experience standpoint.
* “The Gaming Room” development teams will need to be able to implement the system facilitated by CTS and continue to operate and maintain it.

**Technical Requirements**

* Have the ability to have one or more teams involved.
* Each team has 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 game can exist in memory at a time. Accomplish by creating unique identifiers for a game, team, or player.

**Functional requirements**

* Able to render images from a large library of stock images for users to guess.
* The full game consists of four rounds lasting one minute each.
* Drawings rendered steadily (as opposed to incrementally over a given time).
* Images take 30 seconds to render fully.
* When the current team fails to guess the item, other teams will get 15 seconds to guess the image.

## [Design Constraints](#_2et92p0)

* Cost of continuous maintenance and management of new web based distributed environment.
* Cost of hardware and software infrastructure to host the game indefinitely.
* Time to fully implement the system streamlined by CTS.
* Hardware limitations of most common devices owned by users.
* Limitations of user web browsing software.
* Security and ethics considerations for storing user data and information.
* Compatibility and functionality issues between operating systems and web browsers such as Windows, Linux, Mac OS, and various mobile platforms.
* The ability of and the time it will take for “The Gaming Room” development teams to understand and implement the system facilitated by CTS.
* Cost of third-party services related to data storage, encryption, etc.

## [Domain Model](#_8h2ehzxfam4o)

Several object-oriented programming (OOP) principles are evident in the given UML diagram. Inheritance is shown by the relationship between the entity class and the game, team, and player classes. Due to the arrow pointing at entity coming from each of the other three classes, it is clear game, team, and player all have an “is a” relationship with the entity class, meaning each are a form of an entity. Also, GameService is associated with the game, team, and player classes. However, this is a “has a” relationship. This means GameService can have a number of games, games will have a number of teams, and teams will have a number of players. From the relationship classification “0…\*” it is shown there can be any number of these entities, meaning that at any time the existence of the entities is optional. Even though a game technically has a preferable number of teams and teams have a preferrable number of players, the software will operate at times when no instances are in use. Furthermore, due to the nature of the software, players can leave at any time and the software should still operate for the remaining players. Polymorphism is another OOP principle demonstrated, because Game and Team classes have constructors that allow object instances to have various forms depending on the information provided at the time of instantiation. Lastly, encapsulation is demonstrated because some attributes and variables are kept private while accessor methods are made public. For example, nextGameId, nextPlayerId, and nextTeamId are all private, while the get methods for these are made public.

## [Evaluation](#_2o15spng8stw)

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | PROS  -Wide range of streamlined developer tools  -Leverages UNIX tools natively  -Simple development, testing, deployment  -Limited memory/processing management control  -Virtual machine compatibility  CONS  -Limited hardware options/expensive hardware  -Limited scalability  -Limited resource allocation capabilities compared to other Operating systems  -Limited customization  Mac OS has a Unix-based architecture and therefore powerful command-line tools, like Linux. It is known for its user-friendly interface and development environment, which can make development easier if the developers are familiar with the operating system. Mac OS systems are expensive but streamlined, and there are limited options for hardware and software compared to Windows. | PROS  -Open source  -Stable and secure  -Powerful and efficient  -Reliable and stable  -Virtual machine compatibility  CONS  -Greater learning curve  -Less user friendly  -Not as popular as other operating systems  Linux can be great for server-side development because it has many open-source software options and is both stable and secure. However, it has a greater learning curve than any other operating system and isn’t as user friendly. | PROS  -Widely used  -User friendly  -Intuitive GUI  -major support for third party tools and technologies  -extensive server hardware and tool options  -Virtual machine compatibility  CONS  -Less cost effective than Linux  -More security threats compared to other operating platforms  -Licensing required on a user basis.  Windows provides a wide range of development tools and supports popular web server solutions while also offering Windows Server platform. However, Windows may not be as stable, secure, and performance may not be as good as Unix-based systems such as Linux. Also, it can be more resource intensive. Windows tools and technologies are some of the most widely used, have ample documentation and resources for using them. Due to the amount of tools and hardware, Windows is a very scalable option that is cost effective as well. | PROS  CONS  -System memory, storage, and computing power limitations  -Limited compatibility  -User interface very different from other platforms  Mobile operating systems have limited hardware resources and lack development tools for server-side development, except for a small number of flagship tablets that might be able to handle this type of work. Furthermore, mobile devices consist of a wide range of operating systems from Windows, Android, Google, MacOS, and many others. |
| **Client Side** | PROS  -Powerful developer tools and strong support for web standards and technologies  -User friendly  -Highly integrated  -Strong support for web standards and technologies  CONS  -Expensive  -Less powerful Hardware  -Limited compatibility with certain tools and platforms  Mac OS has modern web browser support and supports many technologies for client-side development of web applications. It also offers a user-friendly interface and design features that can assist developers in creating client-side components. MacOS is more popular for client-side development than server-side development. | PROS  -Open source  -Customizable  -Cost effective  -Security & Stability  CONS  -Less user-friendly  -Limited support  -Hardware compatibility issues  Linux is more popular for server-side development, however it still offers plenty of tools for both sides of development, and is very stable and secure. However, Linux has a greater learning curve than other operating systems for implementing the systems necessary for client-side development. | PROS  -Compatibility and versatility  -Highly customizable  -Wide range of tools  CONS  -Security vulnerabilities  -Not open source  -Strong support for web standards and technologies  Windows offers a lot of very commonly used tools for client-side development. While it isn’t the most secure, it is still considered safe and has the necessary tools and technologies to perform all aspects of development for the web application. | PROS  CONS  -Limited hardware capabilities  -Major UI  considerations  -Limited compatibility  It is much more difficult to develop client-side components on mobile platforms as opposed to major operating systems due to the limited hardware and software tools, and variance in size and type of device and operating systems. Any other operating platform would be better than mobile operating systems. |
| **Development Tools** | -Xcode IDE for Apple software  Development  -Apache Web Server for delivering web content  -Nginx open-source web server with advanced features  -Atom free and open-source code editor  -Visual Studio multi language IDE  -React JS library  -Vue JS framework for building UI  -Node JS runtime environment  -Ruby on Rails web app framework  -Angular JS framework for complex/dynamic web app development | -Eclipse open-source java IDE  -IntelliJ IDE for java development  -Apache Web Server for delivering web content  -NGINX  Node JS runtime environment  -Django/Python open-source web framework  -Command line tools  -Emacs coding editor | -Microsoft Azure cloud storage, networking  -Eclipse open-source java IDE  -Visual Studio multi language IDE  -Apache Web Server for delivering web content  -VB.NET web app coding language  -ASP.NET web app framework  -Node JS runtime environment  -Django/Python open-source web framework | -Android Studio  -Android App IDE  -Xcode IDE  -AppCode IDE for macOS & Android  React Native JS framework for developing macOS & Android Apps  -Flutter mobile app development framework  -Xamarin cross-platform app development framework  -Firebase mobile & web app development platform |

**"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.**

## 

## 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 is Windows operating system and hardware because it is user friendly, scalable, has cost effective storage and security options, and has a wide range of development tools that are user friendly. Windows development architecture also has many cross-platform solutions. Development of this web based game will be a little less streamlined than a Mac OS system, but far more streamlined than a Linux system. Windows is also the most widely used and therefore has greater support than other OS options. Development of the web-based game will be much easier for developers in a Windows environment with only a minimal increase in cost compared to Linux. It will also be less expensive than a Mac OS development environment.

**Operating Systems Architectures**:

The Windows operating system is a multi-layered architecture that provides a platform for running user/system processes, and interacting with compatible hardware devices. The operating system consists of various layers including the kernel layer and the user layer. For the user to utilize system resources, Windows first switches to kernel mode to verify the access is authorized, then switches back once the access is verified. This is just one layer of security implemented by Windows. The system also separates any bit of user or system code that the computer can execute into a process, and each of these processes has a state. Furthermore, processes can be independent or cooperating and therefore use a message passing system implemented by the OS to pass messages regarding process status and outcomes of calculations needed for other processes. As will be discussed in the storage management and memory management sections, processes can be executed by various CPU types that utilize an array of execution methods like multi-threading and multi-processing. This architecture is very flexible and allows many different types of hardware to be used to get the necessary performance. Almost every component of a Windows system (central processing unit, random access memory, permanent storage, power supply unit, and motherboard) can be replaced when it is either failing or out of date.

1. **Storage Management**:

A good Windows solution for a small, distributed system would utilize cloud storage such as AWS or Google Cloud Services. This solution would be reliable, easy to implement, but scalable at the same time. This system architecture would be accessible using REST APIs, which the back-end framework will access from the server.

1. **Memory Management**:

Multi-programming is used so multiple processes can occur at the same time. For example, as the image is being drawn during the game, the software can accept input or perform other tasks without having to stop drawing the object. This is important because a requirement of the software is the object will be drawn in a steady manner. Process scheduling will also be implemented to separate various tasks (including I/O bound and CPU bound tasks) which will further improve the effectiveness of the data transfer and memory management during the game. Windows also utilizes information sharing/shared memory which can allow multiple users and classes to access the same piece of information. This is beneficial to the web-based game because it means the state of the game can be effectively and consistently communicated to each user. Windows hardware and software also utilize multithreading which will separate processes among the various cores of the processor and run multiple threads concurrently. Lastly, Windows utilizes Paging and Segmentation to decrease memory fragmentation by allocating memory in fixed sized pages and variable sized segmentations that dynamically ensure that sections of free space do not continue to get smaller and smaller as time goes on.

1. **Distributed Systems and Networks**:

The state of the game will be communicated to each player through the client/server connection, using front-end and back-end technologies. This means the back-end framework will control the logic that transfers the data from the server to the player, and the front-end framework will determine how the viewer will view that information, and what they can do with it. In terms of formatting, the front-end framework should use proper formatting techniques to ensure the game will operate normally for any device, so the user experience is consistent. This can be accomplished through responsive design, and APIs that can cater to the layout to specific devices, if necessary. The server will need to have a reliable connection to the internet and should be monitored and maintained constantly. Windows Server can accomplish this using tools such as performance monitor, event viewer, and various backup/recovery tools. Furthermore, the network will be made robust by choosing an appropriate network topology, implementing network security measures, and providing redundancy to mitigate the effects of a hardware outage.

**Security**:

User data encryption and two-factor authentication/authorization can be implemented at various levels in a Windows development environment. A well-established encryption library should be used such as OpenSSL that can handle encryption and decryption. This will make it easy for the development team to implement and will provide strong security protocols. Similarly, a third-party authentication service can be used that interacts with the back-end framework for the “Draw it or Lose it” game. In addition to the network recommendations above, the network will need to have security protocols as well, such as a firewall and IDPS to detect attacks in real time. Many Windows platforms use dual-mode operation, which ensures proper execution of operating system and user operations by switching to kernel mode when certain OS system calls are made by user programs.