

Draw It or Lose It

# **CS 230 Project Software Design Document**

Version 1.0

## 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 |
| --- | --- | --- | --- |
| 1.0 | 02/19/2023 | Aeriel Denmark | Completed section Recommendations |

## [Executive Summary](#_sbfa50wo7nsh)

The Gaming Room wants to develop a web-based game that serves multiple platforms based on their current game, Draw It or Lose It, which is currently available in an Android app only. The application will render images form a large library of stock drawings as clues. A game consists of four rounds of play lasting one minute each. Drawings are rendered at a steady rate and are fully complete at the 30-second mark. If the team does not guess the puzzle before time expires, the remaining teams have an opportunity to offer one guess each to solve the puzzle with a 15-second time limit.

## Requirements

* Provide a game with the ability to have one or more teams involved
* Each team will have 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 the game can exist in memory at and given time by creating unique identifies for each instance of a game, team, or player.

## [Design Constraints](#_2et92p0)

* Each team needs multiple players
* Game and Team names must be unique to allow users to check whether the name is in use or not
* The ability to run on multiple platforms and not just Android
* One or more team must be involved
* Only one instance of the game can exist at a time

## [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 class file creates a relationship between the Game, Team, and Player class. As a result, they will inherit their information from the Entity class. Therefore, each class will share variables, methods, and accessors. This makes the Entity class a superclass. The GameService class has a reference of Games, Games a reference of Team, and Team a reference of Player. This UML diagram demonstrates the object-oriented programming principle inheritance.

**"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** | Stability, security, and user-friendly interface; provides a full suite of server-side tools and technologies like Apache web server, PHP, and database systems; reliable for running mission-critical applications; security reduces the risk of hacking and malware attacks; compatible with wide range of software for easy integration with existing infrastructure; more expensive than Linux; proprietary hardware and software causes limitability with customization | Open-source for more flexibility and customization; robust nature makes it ideal for stability, security and reliability; versatile and supports a wide range of hardware architectures and software packages; free and affordable option for hosting web applications; has the ability to handle large amounts of traffic and easily scalable; large and active community of developers and users; has a learning curve for users not familiar with interface; limited user-friendly tools; limited proprietary software support | Support across multiple programming languages; large number of tools for server management and administration; Graphical User Interface (GUI); Adopted and supported widely; low learning curve for maintenance and setup due to familiarity; compatible with large number of web-based software applications; high cost compared to open-source alternatives; more susceptible to malware and security threats; not as many customization options as open-source alternatives | Various sensors that can assist with additional functionality and make the user experience more enjoyable; Touchscreen interface which can be more intuitive and user friendly; portability gives user the ability for access on the go and provides convenience; less expensive than traditional computers on the market; ability to access quickly and easily; processing power limited; screen sizes vary and are often smaller; rely on internet connection |
| **Client Side** | Range of development tools and resources to make it easier for developers to deploy; reliable and secure for hosting web-applications; widely used in creative industries; high-performing hardware; Unix-based architecture; user-friendly graphical interface; Hardware typically more expensive than other hardware options; user base smaller compared to other operating systems; not widely supported compared to other operating systems | Widely supported by a large community of developers and users and easy to troubleshoot; cost-effective and the ability to use without license fees; stable and reliable operating system for applications with high availability and up-time; know for robust security features to protect against cyber-attacks and data theft; open source and free, making it more accessible; many not be compatible for some proprietary software; steep learning curer for beginners | High level of security and protection against malware and viruses; widely adopted and easy to find support and resources; compatible with large range of software and development tools; robust security features; user-friendly interface; vulnerable to malware and viruses; performance issues with older hardware; requires regular software updates for security and maintenance | Built in connectivity options to allow user the ability to connect to the internet on the go; Touchscreen interface which can be more intuitive and user friendly; portability gives user the ability for access on the go and provides convenience; large user base to reach a wider audience for web-based applications; convenience to access anywhere at any time; limited compatibility and optimization due to varying resolutions, operating systems, and screen sizes |
| **Development Tools** | Server-side: Ruby on Rails, Python (Django, Flask), PHP, Node.js, Java  IDE’s: PyCharm, RubyMine, Visual Studio Code, PHPStorm  Client-side: CSS, JavaScript, HTML  IDE’s: Sublime Text, Atom, Adobe Dreamweaver, Visual Studio Code  Tools: Git, Slack, Terminal, Homebrew | Server-side: C++, Python, Java, PHP, Perl, Go, Ruby, JavaScript  IDE’s: Visual Studio Code, RubyMine, Eclipse, PyCharm, Komodo  Tools: Apache, Flask, Node.js, Django, Ruby on Rails  Client-side: CSS, Typescript, JavaScript, HTML  IDE’s: Visual Studio Code, Atom, Sublime Text  Tools: React.js, Angular.js, SASS | Server-side: C#, Java, PHP, Python, Node.js  IDE’s: Visual Studio Code, Eclipse, NetBeans  Tools: SQL  Client-side: HTML, CSS, TypeScript, JavaScript  IDE’s: Visual Studio Code, Atom, Sublime Text  Tools: React.js, Angular.js, Vue.js | Server-side: Node.js, Python (Django, Flask), .NET, PHP, Ruby on Rails, Java  IDE’s: Visual Studio Code, Eclipse, Sublime Text  Tools: Firebase, Git, AWS, Docker  Client-side: Swift, React Native, C#, Kotlin  IDE’s: Visual Studio Code, XCode, Android Studio  Tools: Google Play Console, AWS Mobile Services, Firebase, Fabric, TestFlight |

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**:

To allow The Gaming Room to expand Draw It or Lose It to other computing environments, an appropriate operating platform would be a cloud-based platform that offers cross-platform compatibility. Some factors to consider when selecting a cloud-based platform for this purpose are cross-platform compatibility, scalability, security, and cost-effectiveness. Based on these factors, a cloud-based platform such as Microsoft Azure, Amazon Web Services (AWS), or Google Cloud Platform (GCP) would be an appropriate choice. These platforms offer cross-platform compatibility, scalability, security, and cost-effectiveness, and can provide the necessary infrastructure and tools for hosting and deploying Draw It or Lose It to new computing environments. These platforms also offer a variety of features and services that can be leveraged to enhance the game, including data analytics, artificial intelligence, and machine learning.

1. **Operating Systems Architectures**:

Amazon Web Services (AWS) is a cloud-based operating platform that offers a wide range of services, including compute, storage, databases, analytics, and security. The architecture of AWS is based on a global network of data centers, which are connected by a high-speed network. This allows users to deploy their applications in a variety of regions around the world, to minimize latency and improve performance. The platform uses the AWS Elastic Beanstalk, which is a service that simplifies the deployment and scaling of web applications on the platform.

Google Cloud Platform (GCP) is a cloud-based operating platform that offers a wide range of services, including compute, storage, databases, analytics, and machine learning. The architecture of GCP is based on a global network of data centers, which are connected by a high-speed network. This allows users to deploy their applications in a variety of regions around the world, to minimize latency and improve performance. The platform uses the Google Kubernetes Engine (GKE), which is a service that simplifies the deployment and management of containerized applications on the platform.

Microsoft Azure is a cloud-based operating platform that offers a wide range of services, including virtual machines, databases, analytics, and storage. The architecture of Azure is based on a global network of data centers, which are connected by a high-speed network. This allows users to deploy their applications in a variety of regions around the world, to minimize latency and improve performance. The platform uses the Azure Fabric Controller, which is a distributed system that manages the deployment and scaling of applications on the platform.

1. **Storage Management**:

For Amazon Web Services (AWS), use Amazon Simple Storage Service (S3) which is a storage management system that can be used with AWS. It provides a simple and scalable object storage service, with features such as versioning, lifecycle policies, and access control. S3 is designed for high durability and availability, with data automatically distributed across multiple data centers.

For Google Cloud Platform (GCP), use Google Cloud Storage, which is a storage management system that can be used with GCP. It provides a simple and scalable object storage service, with features such as versioning, lifecycle policies, and access control. Cloud Storage is designed for high durability and availability, with data automatically distributed across multiple data centers.

For Microsoft Azure, use Azure Blob Storage, which is a storage management system that can be used with Microsoft Azure. It provides a simple and scalable object storage service, with features such as tiered storage, lifecycle policies, and access control. Blob Storage is designed for high durability and availability, with data automatically distributed across multiple data centers.

1. **Memory Management**:

When an application such as Draw It or Lose It is launched, the operating platform allocates memory to the application, typically in the form of virtual memory. The platform manages the allocation of memory to different applications, ensuring that each application has the necessary resources to run efficiently. When the application Draw It or Lose It runs, it generates temporary data that is no longer needed after a certain period. The operating platform uses a garbage collection mechanism to identify and free up memory that is no longer needed. This helps to prevent memory leaks and ensure that the application runs efficiently. The operating platform continuously monitors memory usage by the application, identifying any issues such as memory leaks or excessive memory usage. This helps to prevent the application from running out of memory, which can lead to crashes or other errors.

1. **Distributed Systems and Networks**:

To enable communication between various platforms for Draw It or Lose It, a distributed software architecture can be used. This approach allows for the application to split into smaller components that run on different devices and communicate with each other over a network. This allows the application to be more scalable, fault-tolerant, and easily deployable. The network infrastructure that connects the devices should be reliable and have sufficient bandwidth to support communication between the different platforms. The network should also be configured with appropriate security measures to protect against unauthorized access or data breaches. Draw It or Lose It can be built using a microservices architecture, where each feature of the application is built as an independent service that communicates with other services over a network. Each service can be deployed on a different platform, and communication between the services can be achieved using APIs or message queues. In a distributed system, the components and services have dependencies on each other. It is important to carefully manage these dependencies to ensure that the system functions properly. Load balancers can be used to ensure that the workload is evenly distributed across the services, and that no single service is overloaded. Draw It or Lose It can also be built using a service-oriented architecture, where the application is divided into services that can be accessed over a network using standardized protocols such as SOAP or REST. This allows the application to be easily integrated with other platforms, devices, or applications. To ensure that the distributed system and network connectivity is functioning properly, it is important to monitor and test the system regularly. This includes testing for connectivity, outages, and network latency. Monitoring tools can be used to detect any issues and alert the team before they become major problems.

1. **Security**:

When considering security measures for the application, several factors to consider are authentication, data encryption, compliance, access control, and secure communication. To ensure that users accessing the application are authorized, user authentication mechanisms should be implemented. Azure, AWS, and GCP all offer authentication and identity management services. Azure offers Azure Active Directory (AAD), AWS provides Amazon Cognito, and GCP offers Cloud Identity. To protect user information on the platforms, data should be encrypted both at rest and in transit. Microsoft Azure, AWS, and GCP all offer encryption features that can be used to encrypt data at rest and in transit. Azure offers Azure Storage Service Encryption for blob storage, while AWS offers server-side encryption for S3, and GCP offers encryption at rest for Cloud Storage. The recommended operating platforms, Microsoft Azure, AWS, and GCP, all provide compliance certifications for various regulatory standards. Azure is certified for SOC 2, HIPAA, and PCI DSS, AWS is certified for SOC 2, HIPAA, and PCI DSS, and GCP is certified for SOC 2, HIPAA, and PCI DSS. Access control mechanisms can be used to limit access to user data to authorized users only. Microsoft Azure, AWS, and GCP all provide access control mechanisms to control access to resources. To protect user information in transit, secure communication protocols such as HTTPS or SSL/TLS should be used. Microsoft Azure, AWS, and GCP all support secure communication protocols. Azure supports SSL/TLS for web services, AWS provides SSL/TLS for Elastic Load Balancer, and GCP offers HTTPS for Cloud Load Balancing.