The Gaming Room

Draw It or Lose It

# Draw It or Lose It

Version 2.0

## Table of Contents

[Software Design Template 1](#_Toc21086455)

[Table of Contents 2](#_Toc21086456)

[Document Revision History 2](#_Toc21086457)

[Executive Summary 3](#_Toc21086458)

[Requirements 3](#_Toc21086459)

[Design Constraints 3](#_Toc21086459)

[Rationale 3](#_Toc21086460)

[Evaluation 4](#_Toc21086459)

[Domain Model](#_Toc21086459) 6

[Design Constraints 3](#_Toc21086459)

[Design Constraints 3](#_Toc21086459)

## Document Revision History

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0  2.0 | 04/14/24  04/17/2024 | Grace Rodriguez  Grace Rodriguez | Executive Summary, Requirements, Constraints, Rational, Description, Evaluate Operating Platforms  Architecture Recommendation |

## Executive Summary

The Gaming Room is expanding their market reach for their Draw It or Lose It app game by implementing a web-based version. The goal is to increase their profit margin by reaching clients in multiple platforms and will be evidenced in this document.

## Requirements

1. Web-based Accessibility:

It is a requirement for the game to be accessed on the web.

1. Cross-platform Compatibility:

The game must be compatible with other devices and operating systems.

1. Code Reusability.

Ability to reuse code across multiple contexts with little or no modification to save time.

1. User Interface Design:

Game experience must be the same in all platforms and devices.

1. Team Participation and Player Assignment:

Each game should allow different teams with members to be added and created.

1. Assignment of Unique Names for Teams and Game:

Each team and members of the team assigned should have unique names which are unique.

1. Single Instance Identifier Limit and Unique Identifiers:

Only one team can make a move at a time.

1. Testing and Debugging Tool.

Examines contents and properties to locate problem issues.

1. Web and App Performance and Optimization

Compresses images. Minimize HTTP requests. Implement caching. Optimizes code. Monitor continuously.

1. Memory Storage.

Ability to hold files.

1. Cloud Based Gaming.

More convenient and requires less set up.

## Design Constraints & Rationale

1. Multi-Platform Distribution:

The game will be played by multiple players using different platforms.

1. Memory and storage constraints:

The game uses a vast number of images that need to be accessible in real time by team members using different platforms.

1. Unique Names:

Team names and member names must be unique to a user. This ensures privacy and prevents confusion for every player.

1. Single Game instance:

Each team that is playing needs to take turns guessing. Only one team and its members should be able to make guesses.

Evaluation.

Operating System Structures

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Development  Requirements: | Mac | Linux | Windows | Mobile Devices |
| Server Side | Characteristics:  Stable and reliable server environment.  Advantage:  Web hosting capable through Unix.  Server based deployment :  Centralized  Licensing cost:  Median  Weakness:  Software options are limited compared to Linux. | Characteristics:  Efficient and powerful.  Advantage:  Cheap.  Server based deployment ?  Licensing cost: Cheap  Weakness:  Complicated set-up. | Characteristics:  Most widely used. Supports web hosting.  Advantage:  Easy interface navigation.  Server based deployment ?  Licensing cost:  Comparable  Weakness:  Costly. | Characteristics:  Access to web app but requires cloud hosting.  Advantage:  Scalable through cloud.  Server based deployment ?  Licensing cost:  Depends but costly  Weakness:  Server hosting limits. |
| Client Side | Considerations:  Cost: Relative  Time: Relative  Expertise: Mac expert  Web Technology:  1.HTML  2.CSS  3. JAVA  Safari Testing required | Considerations:  Cost: Cheap  Time: Timely  Expertise: Linux expert  Web Technology:  1.HTML  2.CSS  3. JAVA  Cross compatibility testing required. | Considerations:  Cost: Low  Time: Low  Expertise: Adequate  Web Technology:  1.HTML  2.CSS  3. JAVA  Windows and browser testing. | Considerations:  Cost: Costly  Time: Costly  Expertise: Mobile app expert  Web Technology:  1.HTML  2.CSS  3. JAVA  App testing |
| Developmental Tools | Languages and Tools:  1.HTML  2.CSS  3. JAVA | Languages and Tools:  1.HTML  2.CSS  3. JAVA | Languages and Tools:  1.HTML  2.CSS  3. JAVA | Languages and Tools:  IOS(Swift)  JAVA |

File System For Data Collection

Data Collection Servers (DCS) and Profile Cache Servers (PCS)

Data Integration Library (DIL)

Inbound Server to Server

Log Files

Functions of Memory and Storage Management

Data Collection Servers (DCS) and Profile Cache Servers (PCS)

* Data Integration Library (DIL)
* Inbound Server to Server
* Log Files

Memory Management Techniques:

Static memory allocation.

This is where memory for variables and data structures is reserved at compile time before the program runs. Memory blocks size and location are fixed and can’t be changed. Need for allocation and deallocation of memory is avoided.

Dynamic memory allocation

Dynamic memory allocates memory at runtime upon program request. Allows for flexibility and efficiency.

Memory pooling

Divides memory /storage logically to process a job or group.

Garbage collection (GC)

Memory recovery in C# and Java. Automatically frees up space that the program no longer needs.

Custom memory allocators

In C++. Controls memory allocation.

Four main memory management techniques:

1. Paging- Frames where secondary memory is stored. From secondary memory the process iare shifted into RAM whenever required.
2. Swapping- “Best technique for memory management because it provides the most efficient use of system resource” (Shiksha.com). Take out and take in the process from time to time because RAM have limited space. Makes free spaces for other processes and later swapped back to main memory.
3. Segmentation-Process are divided into segments.
4. Compaction- A two step process where all pages not in use are copied into one large contigiuos area. Pages in use are then written into freed space.

Distributed System and Network:

A Distributed system is any environment where several computer devvices are working on a variety of tasks and components spread across anetwork. Example SaaS solutions. Distributed systems are used because the work load is too much for one single device.

TYPES:

1. Client Server Systems

Traditional and simple . Composed of multitude of networked computers with a central interaction to a central server for data storage, processing etc.

1. Peerto Peer networks

One software being run by thousands of computers through distributed workloads.

1. Cell Phone networks

Handset workload sharing, switching systems and internet-based devices.

Domain Model

The diagram of “Draw It or Lose It” game application core is Entity. Class and serves as the superclass for all the other components.

The Game, Team, and Player classes are subclasses. These extends from the parent class Entity. A game is composed of multiple teams and each team is composed of individual players.

The GameService class has a composite relationship with Game class.

UML DiagramA diagram of a computer

Description automatically generated

## Recommendation

1. **Operating Platform:** Considering that there is no cost restraints for this project. I recommend using Windows platform. Eventually the time consumed effectively as expertise is not necessary for game development.
2. **Operating Systems Architectures:** Windows NT Kernel. which provides the foundation for the system’s NT Virtual Memory Manager, process and thread management, and I/O subsystem. Above the kernel is the Windows Socket layer, that provide networking services, and the Win32 subsystem, which supports the execution of Win32 applications. The Windows architecture also includes a number of user-mode subsystems, such as the Client/Server Runtime Subsystem (CSRSS) and the Graphics Device Interface (GDI), that provide additional functionality(architecturemaker.com).
3. **Storage Management:** Cloud Storage should be used for overall performance of the game.
4. **Memory Management:** Swapping Memory Management.
5. **Distributed Systems and Networks:** Client Server network will work wonderfully with cloud technology. A central server should hold processes for the game design.
6. **Security:** (Google cloud service provides necessary security for game development already in their system.)

Routine monitoring by for potential hacks should also be considered.

**Q**  Location of main computer should be locked.

**Access should be limited to a few individuals in the company itself.**

**Protect information such as players information and credit card numbers with**

**firewall that only allows data through ports that game needs to operate on. Use of uniques names and team names.**

Make it impossible to cheat on the game. Detection of cheating.

Address security holes top prevent compromising the game by encrypting sensitive data. Ensure client server security through verification. Perform a checksum on the file and server. Correct discrepancies.

Assign packet sequence number.

Hiding IP address is another way of promoting safety and security.

## References

[www.linkedin.com](http://www.linkedin.com)

[Mac Deployment Overview (PDF) (apple.com)](https://www.apple.com/business/docs/site/Mac_Deployment_Overview.pdf)

Jumpcloud.org

[www.architecturemaker.com](http://www.architecturemaker.com)

splunk.com

www.geeksforgeeks.org