<Name-of-Software-Application>

**CS 230 Project Software Design Template**

Version 1.0

**Table of Contents**

**CS 230 Project Software Design Template** 1

**Table of Contents** 2

**Document Revision History** 2

**Executive Summary** 3

**Design Constraints** 3

**System Architecture View** 3

**Domain Model** 3

**Evaluation** 3

**Recommendations** 5

**Document Revision History**

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 11/13/2022 | Geoff Nix |  |

**Executive Summary**

The Gaming Room wishes to expand on its current game, Draw It or Lose It, currently available exclusively on Android, to be available on all platforms. The game will involve one or more teams, each with multiple players. A large library of images will need to be hosted and rendered upon request, and timers of fifteen, thirty, and sixty seconds will be used throughout the game.

**Design Constraints**

Expanding from Android to all available platforms.

Hosting and rendering individual images upon request from a large image library.

Not allowing duplicate values for game and team names.

Only one game instance can exist in memory at any given time.

Implementing timers that switch the game from one team to another upon completion.

**Domain Model**

GameService is the class that holds the list of games, accessors, and mutators for games, players, and Ids. GameService has a zero-to-many relationship with the Game class, as a GameService can have many games. The Game class holds a list of teams and can add a team to the game. The Game class has a one-to-many relationship with the Team class, as a Game can have many teams. Teams hold a list of players and can add a player to the team. The Team class has a one-to-many relationship with the Player class, as a team can have many players. The Entity class keeps track of the current game. The Game, Team, and Player classes inherit from the Entity class, which holds the id and name of each team and public methods for Game, Team, and Player to access these properties.

**Evaluation**

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Fewer virus attacks, strong customer support, efficient performance, and long life.  More expensive, has lower graphics capability and no hardware customization. | Open Source, and free installation. High level of security, a fast-running system that is easy to redesign.  A deeper learning curve and many Windows and Apple-based apps won’t run on Linux. Not usually used for gaming | Microsoft Windows Server products are operating systems licensed, designed, and supported by Microsoft for multiple simultaneous users. Dedicated servers are easy to set up. Excellent tech support, user friendly, utilizes Active Server Pages (ASP) technology and many Microsoft tool interfaces. Require more system resources, has security issues, is not as stable, and is more costly than Linux. | Scalability, affordability, hardware-free, improved speed, and good disaster recovery.  Accessibility causes security issues, data management issues, and bandwidth limitations. |
| **Client Side** | Solid client-side features to host the game. It can host multiple users, does not offer as much variety as the Android platform, is costlier, and has more demanding hardware restrictions. | Interface similar to Windows, able to host multiple users, and currently offers a large gaming library. | Microsoft has five video game consoles, as well as applications (games), streaming services, an online service by the name of the Xbox network, the development arm named Xbox Game Studios and a plethora of other gaming platforms. | Accessible to all smartphones and tablets via App and Google Play Store. Very cost-effective and accessible by virtually everyone although interactions are limited by the battery life of the device. |
| **Development Tools** | IDEs include Xcode, NetBeans, Sublime, Atom, and VSCode, which are free. Git clients are SourceTree and Tower. Apple Developer Enterprise Program annual fee is 299 USD. | Websites can be developed with lesser-known IDEs like Seamonkey, BlueGriffon, and Quantra; they also can be developed in better-known IDEs like Sublime or VSCode which are moderately priced. | Microsoft offers the Windows SDK, which contains various headers, libraries, and tools to build applications using Visual Studio Code; including C++, and various editions of NET Framework for Windows 7 to the current version | Established companies like IBM and AWS offer mobile app development packages. There are also a number of smaller app developer tools companies to choose from with varying costs. |

**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**: We recommend using the Windows server for this game. Windows servers are user-friendly, with an attractive graphical user interface (GUI) and out-of-the-box functionality. In addition, they provide extended support, sustain third-party applications, and require less maintenance.
2. **Operating Systems Architectures**: The Client-Server model is the recommended architecture for this project. This model will allow multiple users to make requests to our central server via the internet. The server can then validate the user and accommodate the data requested, which for our game is images. An identity management service to create a user access token for the applicable identity would then be implemented. This architecture provides adequate security, data management, and flexibility for additional hardware as we enhance the application.

The Microsoft *kernel* implements the core functionality that everything else in the operating system depends upon. When in kernel mode, the program has direct and unrestricted access to system resources and can directly interact with the hardware.

1. **Storage Management**: Storage management for client-server architecture involves the library of images being stored on our servers. Users, or clients, would have direct access to all images via a network request from their PC, laptop, tablet, or mobile device. This architecture will effectively manage, back up, and protect the data. We would recommend at least 8MB of RAM ON PCs that can hold images locally. The image library will live securely on our Windows-based servers, accessible to all users via a network request.
2. **Memory Management**: Short-term memory will be managed with RAM (Random Access Memory. More RAM will increase system responsiveness, enhance frame rates, shorten boot times, and improve other system performance areas. This will efficiently retrieve images and display them to the users quickly. The cache will work along with the RAM, holding small amounts of data to improve response time to the users. Images would be cached in memory and shared across all the game instances. We recommend Windows servers with adequate bandwidth to ensure fast delivery of the images.
3. **Distributed Systems and Networks**: *Draw It Or Lose It* will contain multiple software components on multiple computers, running as a single system connected by a wide area network. This will provide the scalability needed to expand the system and the redundancy for several machines to provide the same service, which will avoid interruptions in service if a system is unavailable. This environment would effectively split up the work, collectively coordinate efforts, and complete tasks more efficiently.

Clients will communicate with the servers via TCP/IP protocol. RESTful communications and principles will be implemented to securely fetch the images. Every interaction will be independent, and each request and response provides all the information required to complete the interaction.

1. **Security**: On the client side, we will use security tools that automatically discover and report on the web and data access, identify client-side security vulnerabilities, and provide specific threat remedies. These security tools will continuously scan and protect the application from unauthorized and malicious attacks. On the server side, we will utilize SSL certificates to establish a secure connection from the client to the server. This component employs data encryption, then validates and authorizes the transaction allowing the encrypted data to be shared.

The Windows Server operating system also includes a set of security components that make up the Windows security model. These components ensure that applications cannot gain access to resources without authentication and authorization.