**The Gaming Room**

# **Software Design Template**

Version 1.0

## **Table of Contents**

Software Design Template……………………………………………………………………………..1

Table of Contents………………………………………………………………………………………. 2

Document Revision History…………………………………………………………………………… 2

Executive Summary……………………………………………………………………………………..3

Requirements………………………………………………………………………………………….. ..3

Design Constraints……………………………………………………………………………………... 3

Rationale…………………………………………………………………………………………………. 3

Evaluation…………………………………………………………………………………………………5

Analysis and Recommendations……………………………………………………………………….9

## **Document Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| 1.0 | 05/15/2022 | Kenneth Paulsen | Added Summary, Requirements and Constraints |

Instructions: Fill in all bracketed information on page one (the cover page), in the Document Revision History table, in the footer, and below each header. Under each header, remove the bracketed prompt and write your own paragraph response covering the indicated information.

## **Executive Summary**

*Our new client, The Gaming Room, wants to develop a web-based game for multiple platforms. This game is based on their current game, Draw It or Lose It which is currently only available as an Android app.*

## **Requirements**

***Business Requirements:***

*Web based game must be based on their current game: Draw It or Lose It.*

***Technical Requirements:***

*App must be compatible on multiple operating devices*

*App will rely on Client/Server architecture*

*App must be compatible to work on multiple web browsers*

## **Design Constraints**

*1.* The *app will need to be developed with web-based framework/programming languages. Given the scope of the project, this will require multiple frameworks/languages.*

*2. Programming language versions and app features would need to be balanced to account for different browser versions (i.e., some functionality on newer browsers isn’t compatible with older browser versions).*

*3.* *For full functionality, the app will require backend and front-end development. This requires either front end/back-end developers and/or full stack developers on the project.*

4. UX/UI designs and functionality will need to be responsive and customized to accommodate different devices. A UI on a mobile device will have different factors to consider compared to a desktop utilizing app.

5.Testing and QA will be larger in scope and more complex due to the breadth of desired compatible operating devices.

## 

## **Rationale**

*1.*  **The *app will need to be developed with web-based framework/programming languages******-*** *The clients require the app to be compatible with multiple operating devices (i.e. desktop, mobile, etc.). For the best results, this requires programming languages that are optimized for each field (web, mobile). Frameworks can also be used to help speed up the development process. Depending on the complexity and scope, this could require one or several frameworks*

*2.* ***Programming language versions and app features would need to be balanced to account for different browser versions -***As new features are added to browsers configurations change and some of these features aren’t backwards version compatible. Implementing similar functionality between browser versions would add more work and time however this would also ensure the app is accessible for an optimal number of people. While the app will never be accessible to 100% of Internet users, finding an optimal balance is important for both technical and business reasons.

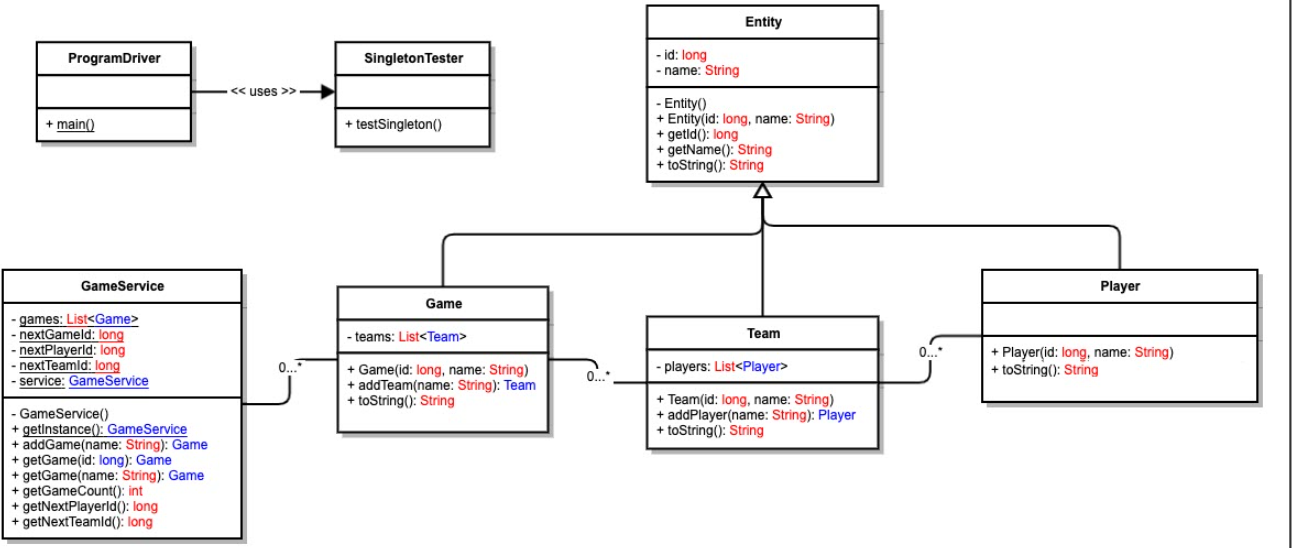
*3.* ***For full functionality, the app will require backend and front-end development –*** The majority of web apps utilize both front end and back-end functionality. Front end primarily focuses on user experience and input. Back end primarily focuses on Database optimizations and API calls between different parts of the application. Based on the desired features, the app will require both front and back-end development. Each side of development has its own special skill sets and knowledge needed to develop effectively. Depending on the capabilities of the development team this could be individuals that specialize in one of these areas or a generalist that can do both.

4. **UX/UI designs and functionality will need to be responsive and customized to accommodate different devices-** Depending on what device is being used will determine the proper layout needed when using the app. For example, mobile phones have constraints desktops do not have such as smaller viewing dimensions, different orientations and *fewer input options. Some UI features supported on a desktop, such as navigation bars, have limited mobile support. These differences will require the development team to customize the UI/UX experiences for each type of device.*

5. **Testing and QA will be larger in scope and more complex due to the breadth of desired compatible operating devices -** The scope and complexity of this project means more resources need to be allocated for testing and QA purposes. Supported features will be different across operating devices as well as different browsers (and their versions). For the app to work across these different scenarios, this requires extensive and rigorous QA, testing and debugging. Failure to do so will result in additional resources being spent over the long-term fixing bugs and features.

**Domain UML explanation:**

The classes Game, Team and Player all share an inheritance with the Entity Class. There are also many multiplicity relationships in this diagram. GameService class has a 0 to many relationships with the Game class, meaning each GameService can have 0 or many instances of the Game class. This exact relationship also occurs between the Game class and the Team class (0 to many teams per Game class) and between Team and Player classes (0 to many players per team). In terms of OOP principles expressed in the diagram we have Inheritance where Game, Team and Player classes are inheriting from Entity class. We also have examples of Encapsulation in the Entity class. The variables id and name are private. They are given initial values when the Entity class is created. However, they can only be accessed using the getter methods in the class and cannot be changed after initialization. There also exists polymorphism through method overloading. For example, the GameService class has a getGame method that can look up a game by either Id or by its name. Since it has several ways of accomplishing the same task, this is an example of polymorphism.

**

## **Evaluation**

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** | Mac utilizes UNIX which is more secure compared to Windows. There is also good tech support from Apple that can help resolve any technical issues. Macs are also higher in quality and have great performance.  However, Mac does have some drawbacks. Mac is not natively supported on anything other than Apple hardware. This means that to set up a Mac OS host will be more expensive since Apple computers are more expensive comparatively. | Linux is free of charge to use. It is also very secure to use and has fewer hardware requirements to utilize. These amount to high-cost savings when hosting with Linux. It also gives the user more flexibility when handling the Linux OS.  However, Linux does have some drawbacks. It is more difficult to maintain/operate. Many of the processes are executed from the command line rather than GUI. While this gives more freedom to the user it is also harder to operate. Some programs are not compatible with Linux and not all Linux versions offer long term support. | Windows is simple to utilize and much of their OS execution mainly relies on a GUI. It also has a great deal of long-term support through Microsoft and can support many different types of applications.  However, it has its drawbacks. It is less flexible compared to other OS such as Linux. It is more prone to security errors/vulnerabilities such as malware. It also has high licensing costs. | Compared to desktop hosting, hosting on a mobile device is much cheaper. The hardware needed is already built into the phone. The app is more secure since it is hosted on the phone itself.  Hosting on a mobile device does have some drawbacks. The performance on a mobile device will not be as productive compared to hosting on a desktop. It also must have a stable internet connection and power. Given the nature of a mobile device, this is not always possible. The connection quality can vary from time to time. Scaling is also an issue since mobile devices are limited by their hardware with very little room for upgrades. |
| **Client Side** | Apple themselves do not offer web hosting. Given Mac has fewer service offerings, and a smaller market share means it's harder to find a Mac Server. However, it is possible to set up an individual server from a hardware perspective, they offer Mac Pro Servers. An entry level model runs $6500. The parts to these servers are all Apple brand, meaning if the server breaks down, it will be costly to fix. | Linux does not have a license cost and is a free open-source platform. For our purposes, Ubuntu Server would be beneficial since it specializes in hosting web sites and can scale appropriately. Best of all, Ubuntu Server is free to use. For additional security and support, it would be wise to purchase Ubuntu Advantage. A standard subscription would be $750 per year for each server. | Windows offers Windows Server for hosting capabilities and specifically designed for servers. Windows Server offers user authentication capabilities, IP address management and filing/storage capabilities.  A standard license costs $1069 and can handle the initial scaling of the application | There are several apps that allow for hosting on a mobile device. However, these can vary in pricing, quality, support and documentation.  Because of these constraints, most companies are switching to hosting applications through the cloud as Platform as a Service (Plaas) rather than hosting on their mobile device. One example of Plaas is Heroku which can be utilized to scale, deploy, and manage the app.  Heroku allows many different features to be added so pricing can vary. For the best run time performance, the Performance M would be the best bet and would cost $250 per month |
| **Development Tools** | For the best results, Objective C and Swift would be the languages utilized for Mac development.  XCode is an IDE that is already installed on Macs and is free to use. However, it is mainly used for software developed exclusively for Apple products. A good cross platform alternative is PhpStorm. This supports front end languages such as HTML, CSS and JavaScript. It also has database support and has numerous plugins available as well. | The most common programming languages used on Linux include C and Python. One of the best IDEs to use for Linux would be IntelliJ IDEA Ultimate. It has many built-in developer and build tools. It also supports many different frameworks giving it cross platform compatibility and mobile support. Pricing is based on individual pricing and can either be paid monthly or yearly. For optimal results, a yearly subscription would be cheaper, and it would be $500 per user annually. | Most development is done through Visual Studio IDE and .NET Core Framework. With Windows, C# and Visual Basic will be the most common language to use. Xamarin would also need to be installed for mobile development. The .NET framework is free to use. Depending on which version of Visual Studio is used, this could be free or cost money. These include Community, Professional and Enterprise. Enterprise would be the best choice for development due to its greater testing and debugging tools. | There are many different IDE’s and frameworks that can be used. One of the most frequently used for mobile app development is React Native. It can develop apps that can work on Web (Windows, Linux, Mac) as well as mobile platforms such as iOS and Android.  It primarily uses JavaScript but also has language modules for Objective C, Java, C++, and Python. React Native is free to use |

**Analysis and Recommendations**

When determining what OS (Linux, Windows, and Mac OS) to use for the web application, I looked at several key factors:

1. Cost
2. Compatibility
3. Ease of Use

**Cost:**

In terms of cost, both Windows and Mac OS require users to pay. Mac OS requires users to purchase a license which depending on the needs can range between $500 - $1000 per server. This doesn’t include the cost of the hardware itself which can cost several thousands of dollars. Meanwhile Microsoft charges $1069 for a standard license per server. This also does not include hardware costs. When it comes to Linux, there are no licensing fees since it is open sourced. The only costs are from the hardware itself. From a cost standpoint, Linux is the clear winner.

**Advantage: Linux**

**Compatibility:**

Mac OS can only be run on Apple designed devices. Any non-Apple devices will not be able to natively run Mac OS. This limits hardware options if Mac was to be used as a server. This also opens several issues down the road especially when it comes to scaling and upgrading servers. Scaling with Mac OS would require purchasing more Apple computers. Upgrading or replacing parts on the servers would depend on whether Apple has the desired part which can be more difficult the older the model. Both Microsoft Windows and Linux are compatible on a wide range of devices. Both Windows and Linux offer backwards compatibility. I would give a slight edge to Linux however since while Microsoft is compatible on many different devices and offers backwards compatibility, users of Linux can alter the source code and tweak it to suit their needs ensuring it can be compatible with virtually any device.

**Advantage: Linux**

**Ease of Use:**

By ease of use I refer to how easy it is to set up and use for the average user. Out of the three, Linux is the hardest to utilize. It is a kernel instead of a full operating system. A typical setup to use Linux would involve taking a Windows computer and repurposing it with Linux. Depending on the needs of the user, this process can become quite complex. To run commands on Linux requires use of the command line which is also not user friendly. This requires more time to set up and can lead to errors down the road if not properly configured. Windows comes with many services already bundled and configured. Any additional packages can be added as need be. Windows also utilizes a GUI rather than a command line making it more user friendly. Mac OS is also user friendly and has an easy setup as well. It is difficult to determine which would have greater ease of use and this would be based on user preferences, but I would give it slightly to Windows.

**Advantage: Windows**

**Recommended OS:**

Linux - The biggest reasons I would recommend Linux is due the flexibility it gives the user as well as its compatibility on a wide range of computers. Costs are reduced since there are no licensing fees and Linux can be utilized on computers of all price ranges. Because of this compatibility, scaling becomes easier to manage since the performance of the OS isn’t based on having the right computer (such as Mac OS). With Linux you also wouldn’t be constrained by needing to have the right replacement part should the need arise such as that with Apple computers. Linux has greater flexibility since the operator can access the source code which is something not possible with Windows or Mac. This allows numerous ways to configure and customize the OS to suit the business needs. The biggest drawbacks to using Linux would be lack of tech support and the greater complexity to install/use. While it is open sourced with many forums and helpful users this support isn’t uniform or as reliable compared to Mac and Windows. If an issue arises, fixing it will rely heavily on how competent the Systems Administrator is. In the wrong hands this could be a major issue using Linux. Since this server would be managed by an IT professional rather than an average user, I feel this issue would be mitigated somewhat.

**Architecture:**

When it comes to the architectural pattern being used, I would recommend utilizing the client-server pattern. This allows a clear distinction of responsibilities between client and server where the server contains the relevant information, and the client utilizes this information. The server would host, deliver, and manage most of the resources and services requested by the client. All the important information can be kept in a central location either in house and/or backed up with a cloud storage provider. This pattern also makes it easy to share important information between the server and client such as game logic and team scores.  This pattern is good for scaling and sharing data among many different platforms. More computers can be added to a client server network to help tackle the additional workload. There are some downsides to this pattern though. If too many users are using the program at once it can cause congestion in the network. Also, if the main server goes down, the entire program can be disrupted. Using this pattern can become costly as more servers are added to scale the growth of the network. It can also require a higher degree of technical complexity to implement. Many of these drawbacks can be reduced somewhat by utilizing a cloud storage provider.

**Memory Management:**

Some of the features of Linux memory management include virtual memory, demand paging, memory allocation and mapping of files to addresses. Virtual memory allows only information needed to be kept in the physical memory address (demand paging). It also comes with a mechanism to protect and control the sharing of data between work processes. This can be handled by the system but also can be configured by the user through the command line.

**Recommended Storage:**

Cloud Storage

Cloud storage allows businesses to delegate handling storage and security to cloud providers allowing them to focus more on the software development itself. Storage costs are usually determined by how many GB of data a business is using. If a business needs to scale up or down, they can increase how many GB’s they need and can be charged accordingly. If this were to be done through traditional means more physical hard drives would need to be purchased. Scaling up can be achieved but if the business needs to scale down then they have hardware not being utilized lying around. With Cloud Storage, the program files can be accessed by any computer connected to the provider regardless of location. Any changes/updates to the files can be synced by the cloud so all users can see those changes. Cloud providers would also be responsible for managing and maintaining the storage devices ensuring parts are repaired and replaced when needed. Also, security could be enhanced through data backup, something that would be a lot more costly to implement in house. With these benefits, cloud computing helps mitigate some of the problems with the client server architecture. Since cloud computing can scale to the needs of the application when a new server needs to be added to the architecture, the cloud provider can handle it. Since backups of the data can be made by the cloud provider if one server crashes with the client-server architecture another server can take its place. With the backups the data remains maintained. Finally, any updates/changes made to the application can be synced with the cloud so all clients will receive the updates as needed.

**Communication:**

Since the game will be browser based and clients will be making multiple requests to the server for resources during the game session, the best communication protocol to utilize would be websockets in conjunction with TCP. Speed will be essential to make sure images are rendered in a timely manner and users can input their guesses quickly. For these purposes, HTTP would be a little bit slow but websockets is faster since it allows bidirectional communication between clients and the server enabling real time data transfer. This communication channel can remain open the whole gaming session or until the server/client exits the program. Web sockets also have the added benefit of being supported on many different web browsers. There are some downsides though. It is more complicated to implement and set up initially compared to HTTP and the web browser must also be HTML5 compliant.

**Security:**

Security for the application can be applied at many different levels. At the network level, an antivirus program can be utilized to prevent harmful viruses from infecting the program. Depending on the application usage, Authentication can require usernames and password credentials to log on. If this is a browser-based game, this feature may be unnecessary however if the game stored user data, such as with leaderboards, then this might be necessary. At the application level, the application can be encapsulated using roles. These roles ensure that only users of a designated role can make changes to the program. Someone with the role of User cannot make changes to the game data or handle database interactions. Another measure at the application level is using data validation. These data validations can come pre-configured such as with annotations that ensure only the right types of data are stored. This can also be handled using standard or customized exception handlers that will halt a program if incorrect data is entered. At the operating level, Linux offers username/password authentication, security audits as well as an access control that only allows certain users access to the file system. Like the roles at the application level, Linux can organize users into different groups with their own access levels.  Permission settings can be applied to files to ensure only desired users can access them. These settings can be configured based on business requirements. There are additional extensions that can be added on as needed such as Buffer Overflow and Memory Corruption checks.

To recap, for this application, I recommend it uses the Linux OS for its web server hosting. Combine this with cloud storage services that can back up the data and scale performance when needed. Given the game requirements, TCP and WebSockets should be utilized for real time updates as needed. Finally at the security level at the bare minimum an antivirus program should be utilized as well as role designation and data validation checks to ensure the application is properly secured.