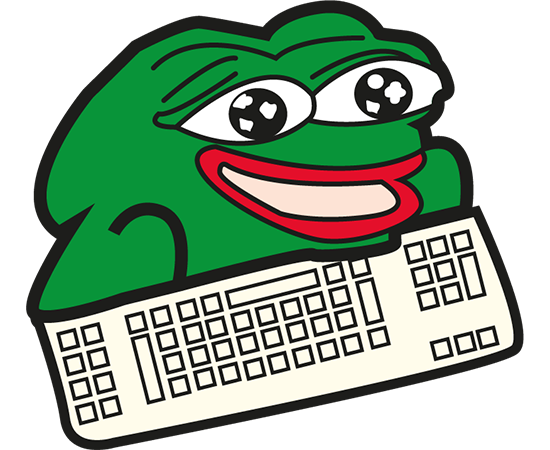
*PEEPOTYPE*



Andrea Sánchez

DAW2

Tutor: Sergi Grau

Contenido

[1. RESUMEN INICIAL 2](#_Toc104175108)

[2. ESTUDIO DE VIABILIDAD 3](#_Toc104175109)

[2.1 ESTABLECIMIENTO DEL ABASTO DEL SISTEMA 3](#_Toc104175110)

[2.2 ESTABLECIMIENTO DE LA SITUACIÓN ACTUAL 3](#_Toc104175111)

[2.2 DEFINICIÓN DE LOS REQUISITIOS DEL SISTEMA 4](#_Toc104175112)

[2.4 ESTUDIO DE LAS ALTERNATIVAS DE SOLUCIÓN Y VALORACIÓN DE LAS ALTERNATIVAS 4](#_Toc104175113)

[2.5 VALORACIÓN DE LAS ALTERNATIVAS 4](#_Toc104175114)

[2.6 SELECCIÓN DE LA SOLUCIÓN 4](#_Toc104175115)

[3. ANALISIS DEL SISTEMA 5](#_Toc104175116)

[3.1 DEFINICIÓN DEL SISTEMA 5](#_Toc104175117)

[3.2 ESTABLECIMIENTO DE REQUISITOS 5](#_Toc104175118)

[3.3 DEFINICIONES DE INTERFACES DE USUARIO 6](#_Toc104175119)

[3.4 ESPECIFICACIÓN DEL PLANO DE PRUEBAS 6](#_Toc104175120)

[4. DISEÑO DEL SISTEMA 7](#_Toc104175121)

[4.0 GUIA DE ESTILOS, USABILIDAD, ACCESIBILIDAD 7](#_Toc104175122)

[4.1 ARQUITECTURA WEB 9](#_Toc104175123)

[4.1.1 DEFINICIÓN DE NIVELES DE ARQUITECTURA: FRONTEND Y BACKEND 9](#_Toc104175124)

[4.1.2 ESPECIFICACIÓN DE ESTANDARES, NORMAS DE DISSEÑO Y CONSTRUCCIÓN 9](#_Toc104175125)

[4.2 REVISIÓN DE CASOS DE USO 10](#_Toc104175126)

[4.2.1 REVISIÓN DE SUBSISTEMAS DEPENDIENDO DE LOS CASOS DE USO 10](#_Toc104175127)

[4.2 REVISIÓN DE CASOS DE USO 11](#_Toc104175128)

[4.2.2 ELECCIÓN DE ALTERNATIVAS DE COMPONENTES Y LICENCIAS 11](#_Toc104175129)

[4.2.4 REQUISITOS DE IMPLANTACIÓN 12](#_Toc104175130)

[4.3 ANALISIS PARADIGMA ESTRUCTURADO / ORIENTADO A OBJETOS 13](#_Toc104175131)

[4.4 DISSEÑO PARADIGMA ESTRUCTURADO / ORIENTADO A OBJETOS 14](#_Toc104175132)

[4.5 PERSISTENCIA DE DATOS: ANALISIS Y DISEÑO 15](#_Toc104175133)

[5. DESAROLLO 16](#_Toc104175134)

[5.1 FORMACIÓN 16](#_Toc104175135)

[5.2 IMPLANTACIÓN DEL SISTEMA Y PRUEBAS 16](#_Toc104175136)

[5.3 NIVEL DE SERVICIOS 16](#_Toc104175137)

[5.4 ACEPTACIÓN DEL SISTEMA 16](#_Toc104175138)

[6. MANTENIMIENTO Y VERSIONES FUTURAS 16](#_Toc104175139)

1. RESUMEN INICIAL

Se trata de una aplicación de mecanografía en la que puedes probar tu nivel, ya sea en solitario o contra alguien. La aplicación te mostrará el WPM, la precisión y los errores. En modo solitario hay diferentes modos, de 60 segundos a 10 segundos y otro modo que no puedes ver lo que escribes. Todos los datos se guardarán para que puedas verlos más adelante.

La aplicación ha sido realizada con JavaScript, ChartJS, EXPRESS, WebSockets y MongoDB también usando LocalStorage.

Es tracta d’una aplicación de mecanografía en la que pots probar el teu nivel, ja sigui en solitari o contra algú. La aplicación et mostrará el WPM, la precisió i els errors. En mode solitari hi han diferents modes, de 60 segons a 10 segons i un altre en el que no pots veure el que escribeixes. Totes les dades es guardarán perquè puguis mirar-les més endavant.

La aplicación ha sigut realitzada amb JavaScript, ChartJS, EXPRESS, Websockets i MongoDB també usant LocalStorage.

It is a typing application where you can test your level, either alone or against someone. The app will show you the WPM, accuracy and errors. In solo mode there are different modes, from 60 seconds to 10 seconds and another mode that you can't see what you write. All data will be saved so you can view it later.

The application has been made with JavaScript, ChartJS, EXPRESS, WebSockets and MongoDB also using LocalStorage.

2. ESTUDIO DE VIABILIDAD

2.1 ESTABLECIMIENTO DEL ABASTO DEL SISTEMA

El proyecto PEEPOTYPE es FrontEnd y también tiene una parte de BackEnd. Los lenguajes que he usado para frontend serian: ChartJS. Para Backend he usado: Websockets, para abrir una sesión de comunicación entre el usuario y el servidor(en este caso express), el cual puedes enviar y recibir mensajes al servidor y recibir respuestas por eventos. El servidor Express que esta alojado dentro del entorno de NodeJS y es un framework, y LocalStorage. Los requisitos que necesitarías para que la aplicación funcione en tu ordenador sería tener

2.2 ESTABLECIMIENTO DE LA SITUACIÓN ACTUAL

La idea del proyecto salió de una página en la que usaba para practicar mecanografía y ver mis resultados. Esa fue mi primera inspiración para hacer este proyecto, yo creo que es una buena idea porque todos pueden hacer servir la página.

Hay diferentes páginas, pero cada una se centra más en un tema en concreto, ya sea solo mejorar la mecanografía con diferentes cursos, o simplemente ver tus resultados, o para competir contra otra gente.

Mi página es una mezcla de todas las anteriores y eso hace que se diferencies de las otras páginas en el mercado actual.

Una aplicación de mecanografía a destacar del mercado sería *MonkeyType,* es la que en el primer párrafo me refería, tiene una idea muy definida y aunque solo se centre en el modo solitario tiene un aspecto visual muy minimalista y formal.

2. ESTUDIO DE VIABILIDAD

2.2 DEFINICIÓN DE LOS REQUISITIOS DEL SISTEMA

Para que tu mismo puedas probar o desarrollar esta aplicación web deberás tener instalados los siguientes lenguajes: MongoDB, Express y Websockets.

En cambio para probar o usar la página web PEEPOTYPE no requerirás de ninguna instalación aparte de MongoDB para que puedas guardar tus valores y ver el de los demás.

2.4 ESTUDIO DE LAS ALTERNATIVAS DE SOLUCIÓN Y VALORACIÓN DE LAS ALTERNATIVAS

2.5 VALORACIÓN DE LAS ALTERNATIVAS

2.6 SELECCIÓN DE LA SOLUCIÓN

El proyecto está programado con JavaScript usando EXPRESS y Websockets, además de MONGO y Local Storage, también aparte he usado ChartJS para crear gráficos ya que es Open Source y funciona con HTML5 Canvas

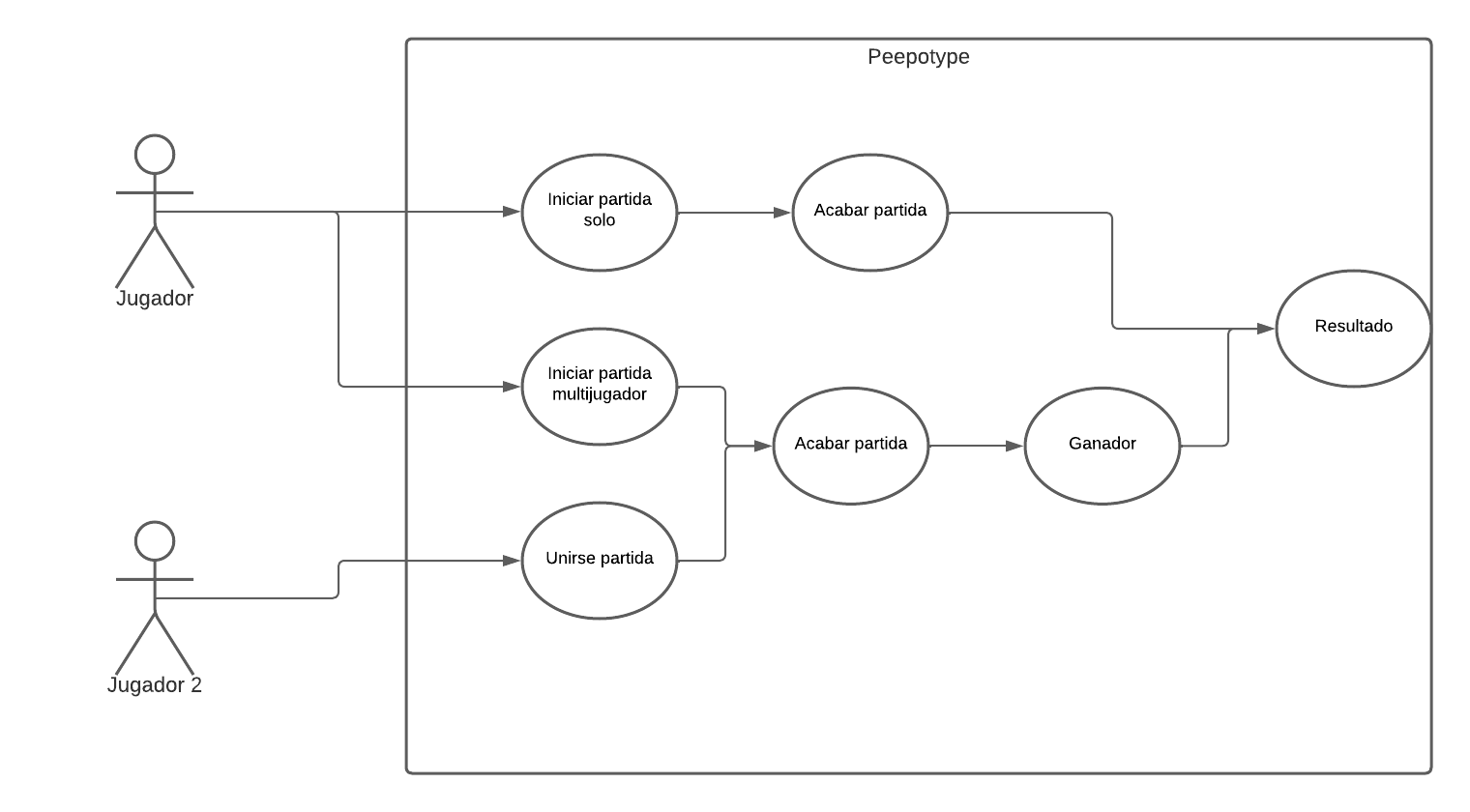
Tomé la decisión de usarlo con JavaScript fue porque tenía pensado usar WebSockets y no sabía si se me podría complicar usando otro entorno de trabajo. Claramente, pensé en desenvoluparlo con otro como Angular pero finalmente me decidí por JavaScript.

3. ANALISIS DEL SISTEMA

3.1 DEFINICIÓN DEL SISTEMA

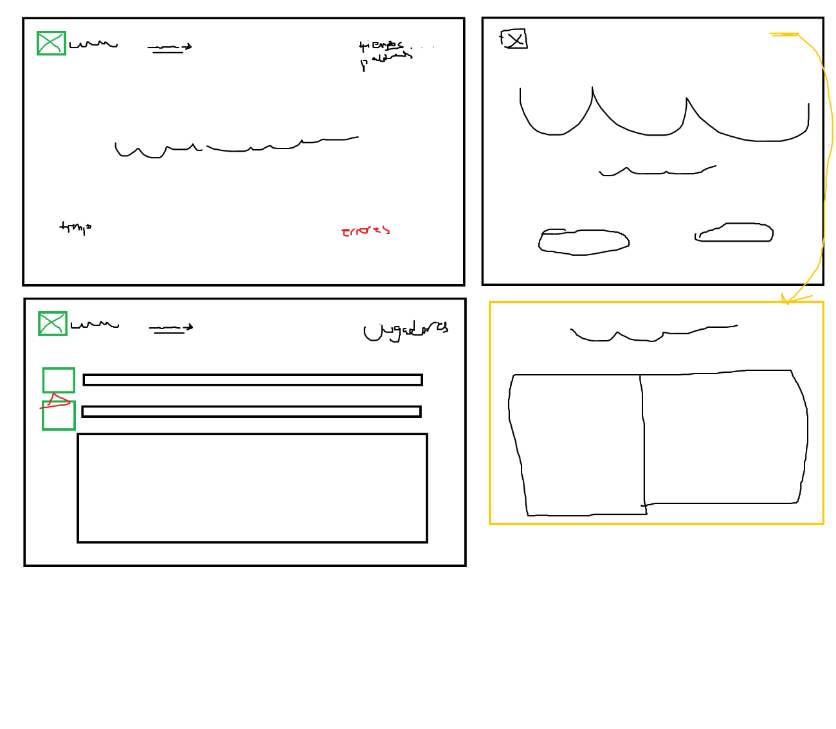
El proyecto sería más enfocado a FrontEnd aunque también tiene una base de datos con Mongo y LocalStorage, bastante simple para poder guardar los resultados de los jugadores que jueguen, LocalStorage está enfocado para el modo solitario de un jugador y utilizo Mongo para el multijugador, el cual guardará también el nombre que el usuario escriba.

3.2 ESTABLECIMIENTO DE REQUISITOS



3. ANALISIS DEL SISTEMA

3.3 DEFINICIONES DE INTERFACES DE USUARIO



Este ha sido el mockup que hice para la página, tenía pensado hacer 4 páginas con un inicio, otro para el solitario, multijugador y la puntuación. Viendo el proyecto final y el mockup creo que he sido fiel al diseño original.

3. ANALISIS DEL SISTEMA

3.4 ESPECIFICACIÓN DEL PLANO DE PRUEBAS

Para la realización de pruebas pedí a conocidos si podían darme una opinión de que les parecía mi página. Cada uno dio distintos puntos de vista o errores que vieron y con ello pude cambiar y mejorar el proyecto a como actualmente está.

4. DISEÑO DEL SISTEMA

4.0 GUIA DE ESTILOS, USABILIDAD, ACCESIBILIDAD

**Logo:**

La imagen representativa de Peepotype sería ese simple texto en mayúsculas, el cual es simple y llamativo. Por otra parte el logo que representaría en si, sería el icono.

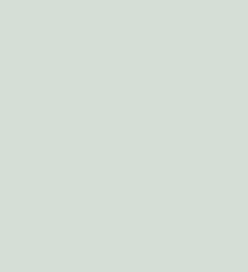
**Colores básicos de la marca:**

Imatge que conté plaça

Descripció generada automàticament #b7edbf

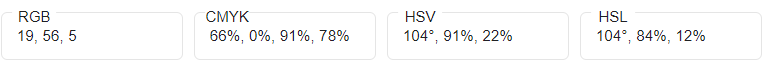


**Colores complementarios:**

 #d5ded6



 #133805



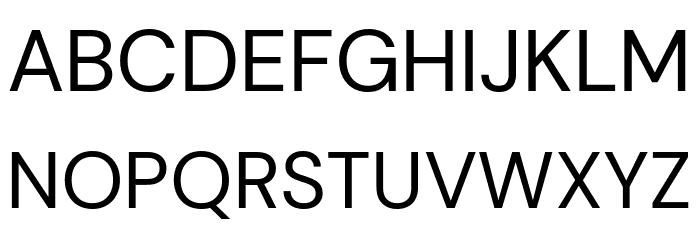
**Tipografías:**

Escogí tipografías sin serifa, de palo seco.

* Monsterrat Italic:



* DMS-Regular:



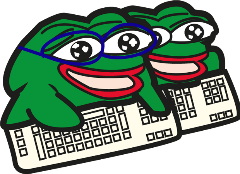
* Radio Canada:

Imatge que conté text

Descripció generada automàticament

**Imágenes e iconos:**

Todas las imágenes son .png, dos de los iconos son creados con illustrator.

Los dos iconos creados por mi:

* **ACCESIBILIDAD**

Una definición rápida de accesibilidad sería diseñar una página web teniendo en cuenta las personas con discapacidades y no influya negativamente su experiencia.

Cosas que he tenido en cuenta

Colores simples y que no molestan a la vista del usuario.

Maquetación lineal y lógica, entender la página web es muy simple y está bien estructurada para ver donde ir en todo momento.

4. DISEÑO DEL SISTEMA

4.1 ARQUITECTURA WEB

4.1.1 DEFINICIÓN DE NIVELES DE ARQUITECTURA: FRONTEND Y BACKEND

Utilice como bien he ido comentando, JavaScript con una parte servidor de Express para poder hacer llamadas al servidor con el cliente, eso era necesario tanto para WebSockets como para MongoDB, el cual hago llamadas con jQuery y Ajax, ya bien para guardar los valores con un POST o con un GET.

Para que funcionará bien todo en la parte del servidor, todos los archivos .js, .json y css están en una carpeta llamada public que luego en el server llamo.

4. DISEÑO DEL SISTEMA

4.1 ARQUITECTURA WEB

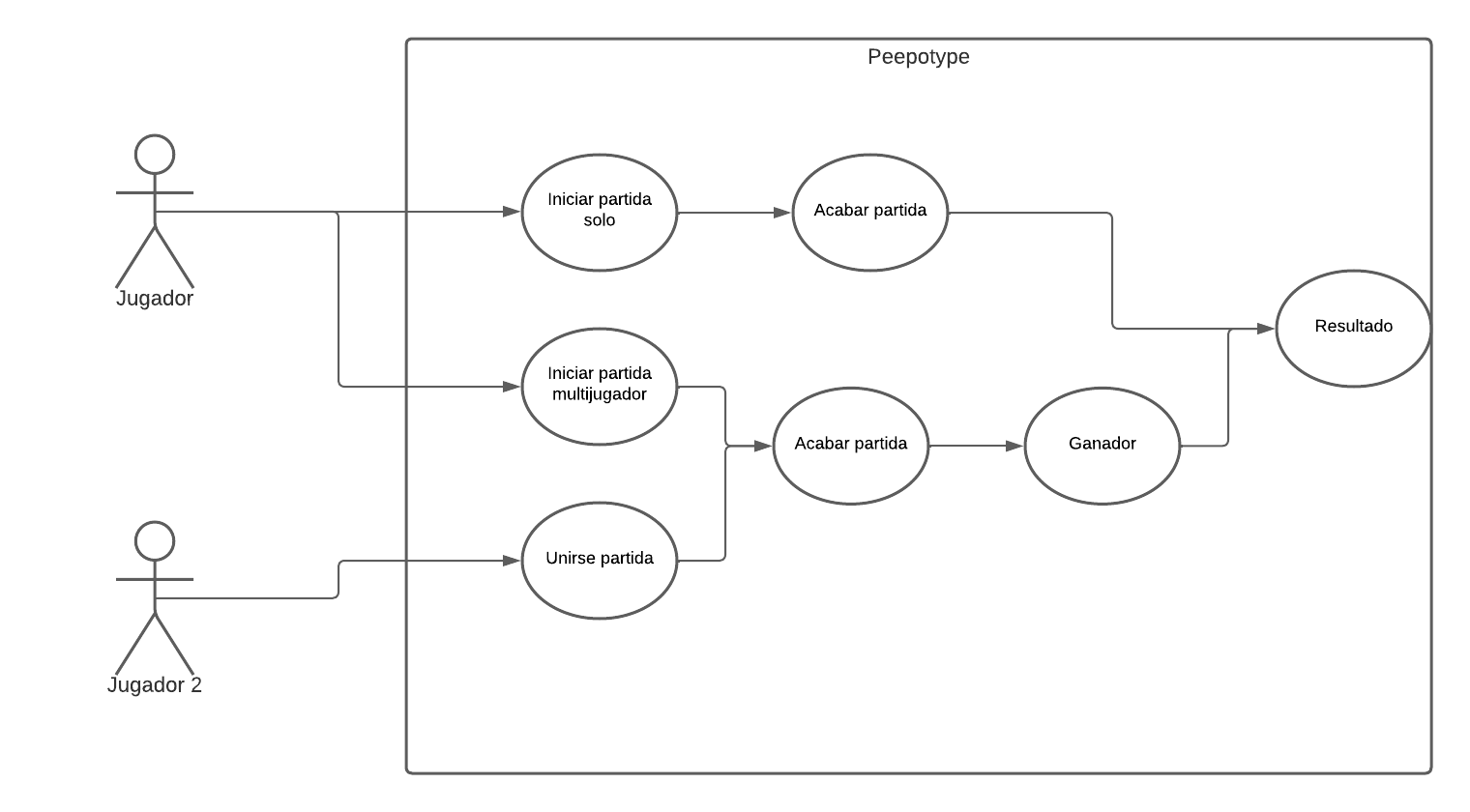
4.1.2 ESPECIFICACIÓN DE ESTANDARES, NORMAS DE DISSEÑO Y CONSTRUCCIÓN

Mi proyecto no está pensado para que cueste dinero en un futuro o se requiera de una suscripción, será todo disponible para el usuario sin ningún coste. No pensé en añadir ninguna interfaz en la que requiera el usuario tener privilegios pagando entonces por eso decidí que todo estuviera al abasto de cualquier persona que use la aplicación.

4. DISEÑO DEL SISTEMA

4.2 REVISIÓN DE CASOS DE USO

4.2.1 REVISIÓN DE SUBSISTEMAS DEPENDIENDO DE LOS CASOS DE USO



Una explicación más detallada sobre el diagrama de casos de uso que he hecho sería que está hecho con actores y tiene el recuadro grande (sistema) con el nombre de PeepoType porque esa sería la página web en la que los actores entran, allá ya se encuentran los casos de uso que me he centrado en cuando empieza la partida (ya sea en solitario o multijugador) y cuando la acaba que lleva a un mismo lado, el resultado.

4. DISEÑO DEL SISTEMA

4.2 REVISIÓN DE CASOS DE USO

4.2.2 ELECCIÓN DE ALTERNATIVAS DE COMPONENTES Y LICENCIAS

4. DISEÑO DEL SISTEMA

4.2 REVISIÓN DE CASOS DE USO

4.2.4 REQUISITOS DE IMPLANTACIÓN

Para que la aplicación funcione para todos los usuarios se debería de deployar en alguna URL, ya sea gratuita o de pago, yo pensé en usar Heroku ya que puedes usar una versión sin pagar y sirve como servicio PaaS y allá puedas usar tu aplicación.

En mi caso, mi aplicación no estaría lista para implantar porque faltaría cambiar un poco de código e implantarlo con Mongo Atlas que es compatible con Heroku y servirá como base de datos, hasta ahora como proyecto inicial usaba mi propio MongoDB y si subierá esta aplicación web la gente no podría guardar sus puntuaciones.

4. DISEÑO DEL SISTEMA

4.3 ANALISIS PARADIGMA ESTRUCTURADO / ORIENTADO A OBJETOS

4. DISEÑO DEL SISTEMA

4.4 DISSEÑO PARADIGMA ESTRUCTURADO / ORIENTADO A OBJETOS

4. DISEÑO DEL SISTEMA

4.5 PERSISTENCIA DE DATOS: ANALISIS Y DISEÑO

5. DESAROLLO

5.1 FORMACIÓN

No requieres de ninguna formación ni documentación inicial para poder entender o usar la aplicación ya que se ha usado lenguajes sencillos y del abasto de cualquier persona. Toda información puede ser encontrada fácilmente en Internet. Tampoco tiene ninguna sistema de administración por lo cual no necesitas información extra sobre eso.

5.2 IMPLANTACIÓN DEL SISTEMA Y PRUEBAS

Se ha testeado con console logs para poder comprobar el error, también poniendo debuggers en el código y con ello se han podido solucionar correctamente.

5.3 NIVEL DE SERVICIOS

No hay ningún sistema de suscripción, demo o pago para esta aplicación web, todo es gratuito y está disponible a todos los públicos sin tener que aportar ningún tipo de servicio monetario.

5.4 ACEPTACIÓN DEL SISTEMA

No ha habido ninguna problema con el cliente entendiendo la aplicación lo que puede que haya sido confusa la parte del multijugador, esa misma, si la aplicación se vuelve más novedosa y famosa se podría mirar a mejorarla, como otros ámbitos de la página web.

6. MANTENIMIENTO Y VERSIONES FUTURAS

Alguna idea del futuro sería que se pudiera usar una URL para que puedan probarlo todo tipo de personas el modo multijugador ya que sigue en “desarrollo” y las puntuaciones se puedan guardar para cualquier persona que lo use. Para ello se necesitaría unas versiones futuras.