Araware Solutions S.L

Calle Cinco de Marzo 8

50004 Zaragoza

ARAWARE

Plan de gestión, análisis, diseño y memoria del proyecto

ARAGOTE

GitHub Organization: https://github.com/anahub99/aragote

Óscar Anadón, 760628

ÍNDICE

1. Introducción **2**

2. Organización del proyecto **2**

3. Plan de gestión del proyecto **3**
3.1 Procesos 3
3.2 Planes 5

4. Análisis y diseño del sistema **9**
4.1 Análisis de requisitos 9
4.2 Diseño del sistema 10

5. Memoria del proyecto **16**

6. Conclusiones **25**

Glosario **27**

Anexo II. Horas empleadas por el equipo de desarrollo **27**

1. Introducción

La aplicación **Aragote** diseñada e implementada por la empresa Araware Solution S.L facilita al usuario poder jugar al tan conocido guiñote desde cualquier teléfono inteligente, necesitando únicamente su ingenio y conexión a internet.

Aragote permite al usuario jugar partidas de 4 jugadores por parejas con amigos o desconocidos. Estas partidas siguen las reglas del guiñote clásico aragonés, tanto en el valor de las cartas como en el conteo de puntos.

Como ventaja de este novedoso formato online, cualquier jugador es capaz de poner la partida en pausa si así lo desea y si cuenta con el beneplácito del resto de participantes, pudiendo retomar en el momento que se desee, nuevamente si todos están de acuerdo.

Además de esto, el usuario tiene la posibilidad de consultar el histórico de partidas, comprobando cuales ha ganado o perdido y quien era su pareja o contrincantes.

El coste inicial del proyecto ha sido tasado en 11.466€, dicho proyecto deberá haber concluido en un plazo de 14 semanas, durante las cuales se hará al menos una presentación intermedia para comprobar el buen funcionamiento del trabajo, acompañada de reuniones periódicas para aclarar dudas y concretar conceptos.

En lo que respecta a la organización de **Araware Solutions S.L**. aparecerá desarrollada en el apartado número 2 de esta misma memoria.

En último lugar se encuentran los requisitos cumplidos asi como las vistas que muestran los detalles de organización del sistema.

2. Organización del proyecto

Con el objetivo de llevar a cabo este proceso la compañía **Araware Solutions S.L** formada inicialmente por 6 empleados ha sido organizada de la siguiente manera.

Óscar Anadón: Director de proyecto, diseñador e implementador de frontend y coordinador.

Jose Maria Hernandez: Diseñador de front-end centrando en la aplicación web.

Sterling Cooper: Diseñador de front-end centrando en la aplicación móvil.

Josh Dell: Implementador de back-end y coordinador entre la parte de front-end y back-end.

John Waio: Implementador back-end, responsable de estructuras y optimización del sistema.

Steve Wall: Secretario de reuniones, responsable de la documentación del proyecto.

3. Plan de gestión del proyecto

3.1 Procesos

3.1.1 Procesos de inicio

A continuación se mostrarán los procesos iniciales establecidos al comienzo del proyecto, los cuales serán identificados con una codificación (**COD_X**) para facilitar su identificación.

COD_0: Se decidirá qué servicio de alojamiento en la nube se va a utilizar, en base a su coste monetario y escalabilidad, principalmente.

COD_1: Se barajearán los posibles lenguajes (o frameworks) para desarrollar el frontend

COD_2: Se barajearán los posibles lenguajes (o frameworks) para desarrollar el backend

COD_3: Elección de un dispositivo para la realización de pruebas, debido a las peculiaridades del proyecto, dicha elección se realizará sobre los terminales de los integrantes del equipo.

COD_4: Selección de emulador para el desarrollo de la aplicación.

COD_5: Elección de entorno virtual de trabajo para backend

COD_6: Elección de entorno virtual de trabajo para frontend

COD_7: Se escogerá una configuración de preferencias en torno a las librerías utilizadas por los desarrolladores de backend para lograr consistencia

COD_6: Se escogerá una configuración de preferencias en torno a las librerías utilizadas por los desarrolladores de frontend para lograr consistencia.

Cabe mencionar que para los desarrolladores tanto de frontend como de backend muy posiblemente les será necesario formarse en determinados lenguajes de programación o empleo de librerías o plataformas. Para ello deberán acceder a cursos, tutoriales o similares, los cuales serán escogidos por ellos personalmente. Esta podría ser considerada también como una decisión de inicio, pero teniendo en cuenta las preferencias, tiempos y estrategias de aprendizaje de cada desarrollador se ha decidido establecerlo como una decisión personal únicamente.

3.1.2 Procesos de ejecución y control

A continuación se hablará de las técnicas y procesos llevados a cabo para el mantenimiento y el correcto desarrollo del proyecto, permitiendo al equipo mantener una comunicación adecuada que a su vez logre cumplir con las pautas preestablecidas por los mismos.

Se establecerá una codificación(**CODE_X**) para la mejor identificación de estos procesos.

CODE_0: Elaboración y establecimiento de una propuesta técnica y económica para el cliente

CODE_1: Elección de los canales de comunicación por parte de backend, por parte de frontend y entre ambos. La finalidad de estos canales es la consulta de dudas simples, confirmación de reuniones etc.

CODE_2: Establecimiento de reuniones (empleando la decisión de CODE_0), a las cuales deberá acudir siempre que se pueda todo el equipo y siempre (de forma imperativa) un miembro de cada división de grupo. Se realizarán aproximadamente con una frecuencia semanal y mediante la solución obtenida por CODE_1

CODE_3: Definición de tareas, complejidad de esta e integrantes que la realizarán. Se decidirán en las reuniones de CODE_2, en caso de que alguien del equipo no acuda se le comunicará posteriormente su labor. Si la tarea de alguno resulta ser mas costosa de lo planificado se decidirá el traspaso a su equipo de algún miembro para agilizar la resolución de este.

CODE_4: Elección de un secretario de reuniones encargado de recopilar la información importante de cada quedada y subirla a la plataforma común del equipo para analizar las actividades comentadas.

CODE_5: Establecimiento de objetivos dinámicos. Decidir cuándo se establecerán las metas a cumplir y los plazos en los que deben cumplirse.

CODE_6: Elección de un director de proyecto encargado de gestionar de manera global el funcionamiento del equipo

CODE_7: Designación del encargado de entrega de los documentos pedidos en cada etapa.

CODE_8: Elección de consenso entre los miembros del equipo para verificar la calidad del producto.

3.1.3 Procesos técnicos

Al igual que con los demás procesos se establecerá una codificación (**CODT_X**) para identificar cada uno de los procesos técnicos facilitando su comprensión.

CODT_0: Se establecerá una batería de pruebas (la cual crecerá con el proyecto) que deberá de pasar cada vez que alguien quiera acoplar funcionalidades nuevas al proyecto. Esta se ejecutará mediante un script que realizará todas pruebas de manera automática con el fin de evitar el fallo humano.

CODT_1: El avance y desarrollo del proyecto se realizará mediante integración continua.

CODT_2: Se comprobará el aspecto y funcionamiento del sistema semanalmente coincidiendo con las reuniones del equipo, se escogerán a uno o dos miembros para que testeen la aplicación de manera casual.

3.2 Planes

3.2.1 Plan de gestión de configuraciones

Para tener una configuración grupal y clara en la que todos los miembros trabajen del mismo modo se emplearán siempre que sea posible los estándares y guías de estilo pertinentes a cada lenguaje usado, todo ello en su publicación más actual. Todas en su versión ofrecida por Google

Así entonces, para backend se ha usado la guia de estilo relativa a **Python** (https://google.github.io/styleguide/pyguide.html), en la parte de frontend, la perteneciente a **HTML** y CSS (https://google.github.io/styleguide/htmlcssguide.html) y finalmente la que explica **JavaScript** (https://google.github.io/styleguide/jsguide.html)

Debido a que **React** no tiene una guía de estilo oficial declarada, los integrantes de frontend encargados de utilizar esta herramienta se pondrán de acuerdo a la hora de desarrollar la aplicación.

En lo respectivo a la gestión de las distintas actividades, no existirán encargados específicos. Cada división de trabajo será el responsable de las actividades pertinentes, junto con la gestión de los backups. Las actividades más generales serán discutidas semanalmente en reuniones a las que deberán asistir todos los participantes (salvo justificación adecuada) de manera presencial o telemática.

El control de versiones se realizará de forma automática mediante el empleo de **Git**.

Debido al pequeño tamaño del proyecto este consistirá en un único repositorio que contendrá todo el desarrollo de la aplicación en sí. Por supuesto este se podrá dividir en directorios según la temática de los ficheros.

Para los posibles cambios a realizar, cada integrante es libre de desarrollar estos en una rama paralela o en local, siempre trabajando con una copia del sistema funcional. En caso de que se tenga una actualización terminada, se pasará una batería de pruebas al sistema con los cambios introducidos, en caso de que pase todas correctamente, se hará commit en el repositorio principal.

Todos los integrantes del equipo serán acreditados para realizar cambios.

3.2.2 Plan de construcción y despliegue del software

Con el fin de que todos los desarrolladores emplean el mismo compilador y librerías, la construcción de la API mediante Django se hará con las siguientes configuraciones

- Se instalará el paquete “virtualenvwrapper” para trabajar en un entorno virtual con las librerías separadas del resto del sistema. A continuación cada desarrollador se creará su propio entorno virtual llamado “lector”
- En dicho entorno se empleará Python 3 como compilador por defecto
- En el momento en el que un desarrollador emplee una librería nueva se añadirá al fichero requeridos.txt.

Coincidiendo con la reunión semanal (aproximada) del equipo, el sistema deberá ser probado con la misma frecuencia, con la finalidad de que cada miembro del equipo comente los problemas y las soluciones que han surgido con las últimas actualizaciones.

Como se ha mencionado anteriormente el despliegue del sistema se realizará en Heroku (PAAS), ya que ofrece un servicio gratuito que permite al usuario mantener un servidor activo durante la mayoría del día.

Por supuesto este nivel de capacidad valdría únicamente para un proyecto didáctico o fases iniciales de la aplicación, a nivel de pruebas. No debería tardarse mucho en contratar un proveedor que ofreciera un alto ancho de banda con niveles de velocidad altos, teniendo en cuenta que para cualquier software orientado a juegos online el “lag” debe ser mínimo para una correcta experiencia y una correcta satisfacción del usuario.

Para evitar posibles fallos humanos y tareas repetitivas el despliegue del software en Heroku será automatizado mediante un script escrito en bash. Dicho script ejecuta un commit en git y actualiza el repositorio de GitHub. Seguidamente actualiza el de Heroku lanzando la nueva versión en producción.

3.2.3 Plan de aseguramiento de la calidad

De manera muy central, la calidad de Aragote se basa en la calidad gráfica del desarrollo de aplicación, como puede ser las tonalidades cromáticas escogidas, el empleo de iconos homogéneos etc.

En cuanto a la primera cuestión, se emplearán a lo largo de toda la aplicación la librería **Material UI**, similar a Bootstrap pero con muchísima más *potencia*. En cuanto al estilo a utilizar, se seguirán los conceptos agrupados en la guía de diseño **Material Design**, desarrollada por Google. Con esto se conseguirá un aspecto sólido y homogéneo que garantizará el buen ver de la aplicación.

Es comprensible que en algo tan subjetivo como es el aspecto de cualquier elemento haya discrepancias entre los miembros del equipo, así pues, para asegurar la calidad de este deberá de haber consenso entre todos los integrantes del equipo, teniendo en cuenta todos ellos que la opinión de los desarrolladores de Frontend deberá tener una ponderación más alta, ya que continuamente consultarán trabajos de diseño, guías, estándares etc.

Esto se realizará coincidiendo con las reuniones semanales (aproximadamente) del equipo.

Respecto a la segunda cuestión, se realizarán test de eficacia, midiendo la velocidad de los procesos de la aplicación, las conexiones que realiza a la base de datos, comprobando tiempos, etc.

Estos resultados serán guardados en documentos junto con la configuración sobre la que se probó, con el fin de comprobar cuál de las configuraciones tiene en el cómputo total la versión más ágil respecto a las necesidades reclamadas. Así el usuario podrá disfrutar de una aplicación fluida que le permita disfrutar de las experiencias del clásico juego del guiñote.

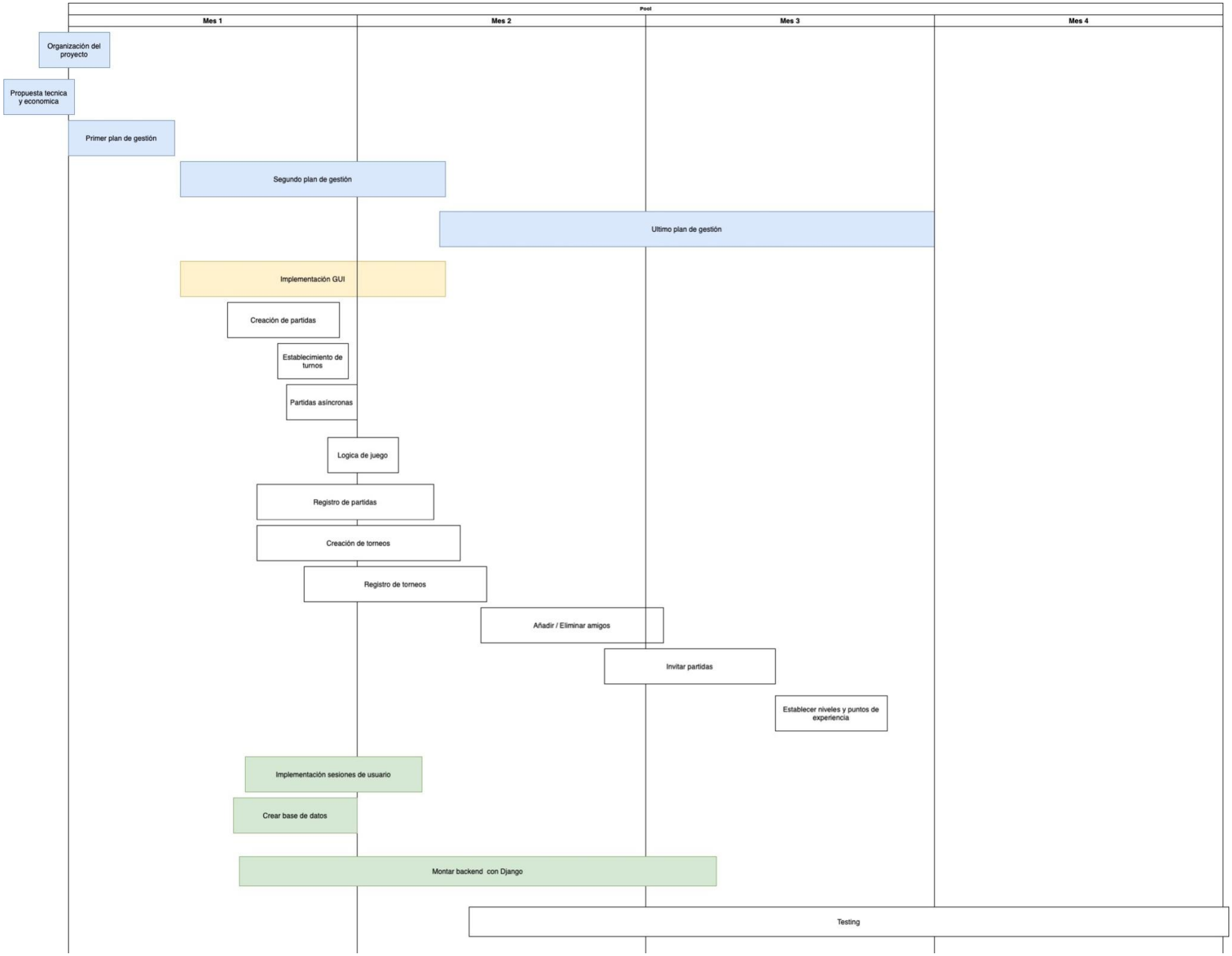
Evidentemente estas pruebas no se realizan semanalmente. En su mayoría se realizan para versiones finales de producción o al introducir grandes cambios o módulos nuevos.

3.2.4 Calendario del proyecto y división del trabajo

Se ha repartido el desarrollo de la aplicación mediante back-end y front-end, los cuales intentarán paralelizar trabajo siempre que sea posible. Se mostrará a continuación en el siguiente diagrama de Gantt.

Se ha dividido el proyecto en dos grupos principales, cada uno de tres personas. Uno de ellos estará dedicado al desarrollo del backend y el otro en el frontend, sin embargo, los miembros de ambos equipos deberán trabajar conjuntamente, ya que hay temas relacionados, y ayudarse debidamente.

A pesar de esta división inicial, en caso de que en uno de los dos ámbitos haya una carga de trabajo mayor, será posible el traspaso de algún miembro del equipo para resolver dicha carga. Todo ello, por supuesto, teniendo en cuenta que es posible que esta persona tenga unos conocimientos más avanzados que las personas ya integradas en el equipo en cuestión.


Pool
Mes 1
Mes 2
Mes 3
Mes 4
Organización del proyecto
Propuesta tecnica y economica
Primer plan de gestión
Segundo plan de gestión
Ultimo plan de gestión
Implementación GUI
Creación de partidas
Establecimiento de turnos
Partidas asíncronas
Logica de juego
Registro de partidas
Creación de torneos
Registro de torneos
Añadir / Eliminar amigos
Invitar partidas
Establecer niveles y puntos de experiencia
Implementación sesiones de usuario
Crear base de datos
Montar backend con Django
Testing

4. Análisis y diseño del sistema

4.1 Análisis de requisitos

Requisitos funcionales

1. El sistema permitirá jugar partidas de 4 personas
2. El juego se adecuará a las reglas generales del guiñote aragonés (https://www.guiñarte.es/reglamentacion)
3. El usuario podrá consultar la última “baza” de sus adversarios y todas las anteriores de su equipo
4. El sistema permitirá pausar partidas cuando un jugador los desee y los otros 3 esten de acuerdo.
5. El usuario podrá abandonar la partida en el momento que lo desee (se da por perdida)
6. El sistema permitirá al usuario modificar el tipo de letra, tamaño y colores de la aplicación.
7. El usuario podrá añadir amigos
8. El usuario podrá eliminar amigos
9. El sistema guardará un registro de las partidas y torneos jugados con información relativa a estos que el usuario podrá consultar
10. El usuario podrá invitar a jugar a un amigo (registrado previamente) concreto.
11. El usuario podrá buscar personas y consultar su perfil
12. El sistema permitirá cambiar la foto de perfil, nombre, correo o contraseña del usuario
13. Un usuario podrá retomar una partida pausada siempre que los demás participantes estén de acuerdo
14. El sistema otorgará puntos a los usuarios según sus partidas jugadas
15. El sistema permitirá al usuario subir de nivel en base a su ritmo de juego
16. El usuario deberá registrarse previamente
17. El sistema mantendrá la sesión del usuario iniciada
18. El usuario podrá cerrar sesión cuando desee (no en medio de una partida)

Requisitos no funcionales

1. El sistema podrá ser desplegado en entornos móviles (**android 5.0** o superior) El usuario está obligado a autentificarse para acceder
2. El usuario deberá de tener acceso a una conexión de red estable (**50Mb** de descarga y **5Mb** de subida)

4.2 Diseño del sistema

Debido a que el proyecto consiste en el desarrollo de una aplicación móvil, se ha decidido emplear una arquitectura en la que el backend le proporcionará una API al sistema móvil.

El frontend será desarrollado para dispositivos con sistema operativo (SO) Android (5.0) o superior. El dispositivo en el que se realizarán las pruebas será un Samsung Galaxy S10, por lo que es posible que para otros formatos de pantalla (uso de “notch”, “lágrima”) la distribución no sea la deseada.

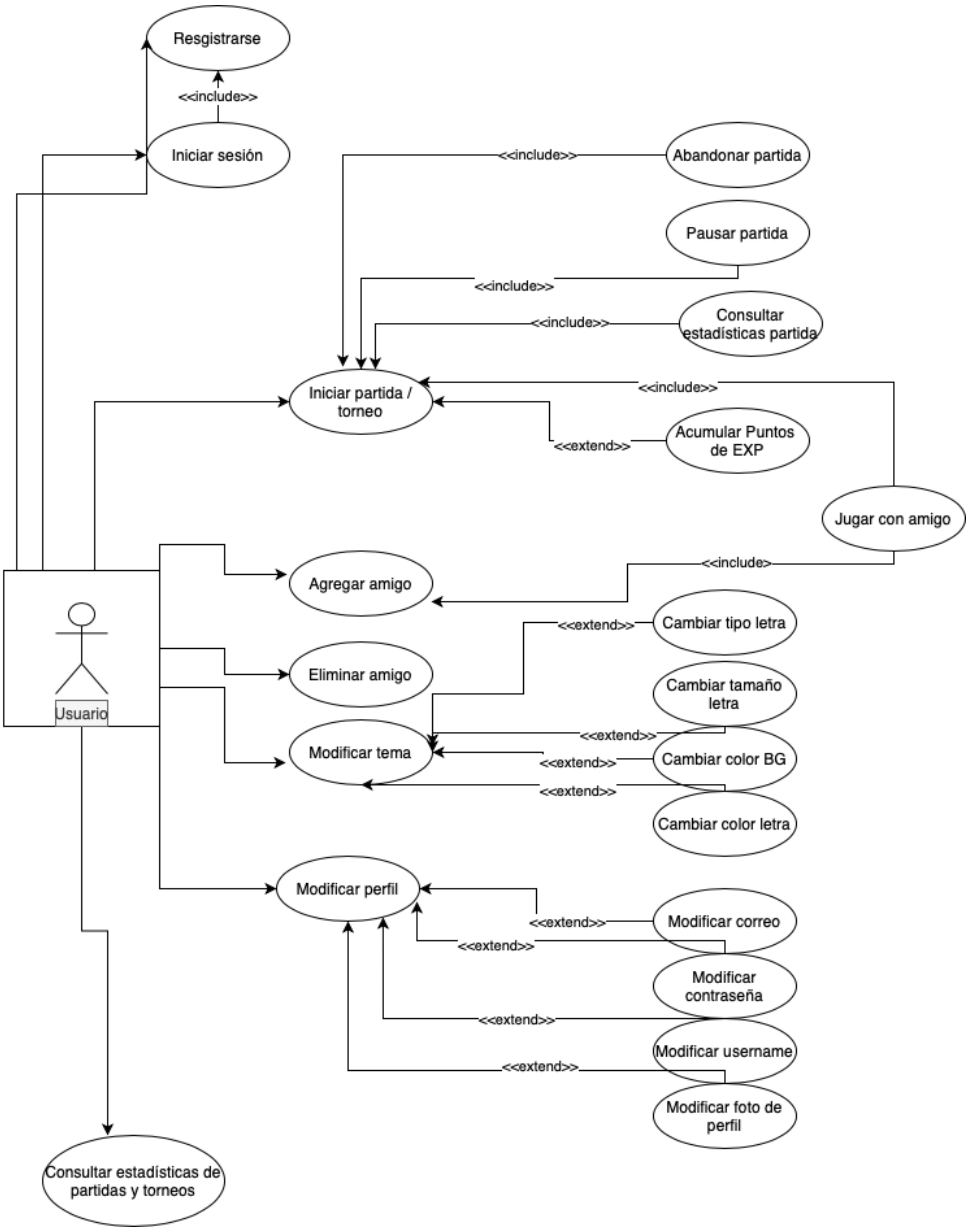
La aplicación va a ser diseñada haciendo uso de React y sus múltiples bibliotecas, así como su lenguaje JSX, considerado una variante de Java Script.

Para todo el tratamiento de los datos, en el backend se implementará una base de datos PostgreSQL, con una estructura ORM junto con el framework en Python Django.

Seguidamente, para el despliegue, se usará Heroku como PaaS, el cual permitirá compilar, ejecutar y desplegar el servicio en la nube.

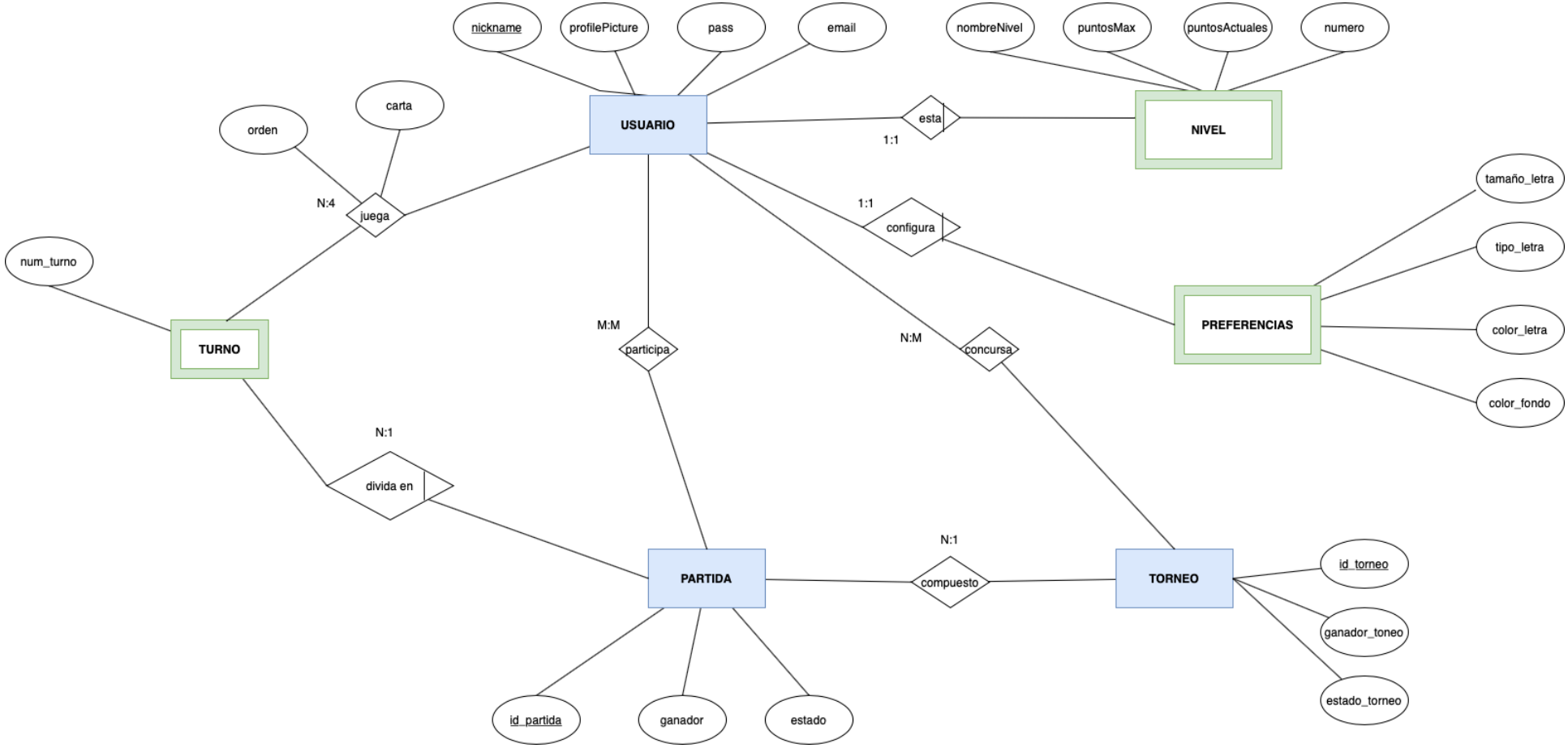
Hay que tener en cuenta que de primera instancia se podrá usar el servicio gratuito de Heroku para pruebas entre los desarrolladores, sin embargo, debido a la velocidad y a la ausencia de “lags” que se espera en cualquier juego online, será necesario realizar un pago para obtener las características necesarias que permitan una jugabilidad fluida

4.2.1 Diagrama de casos de uso

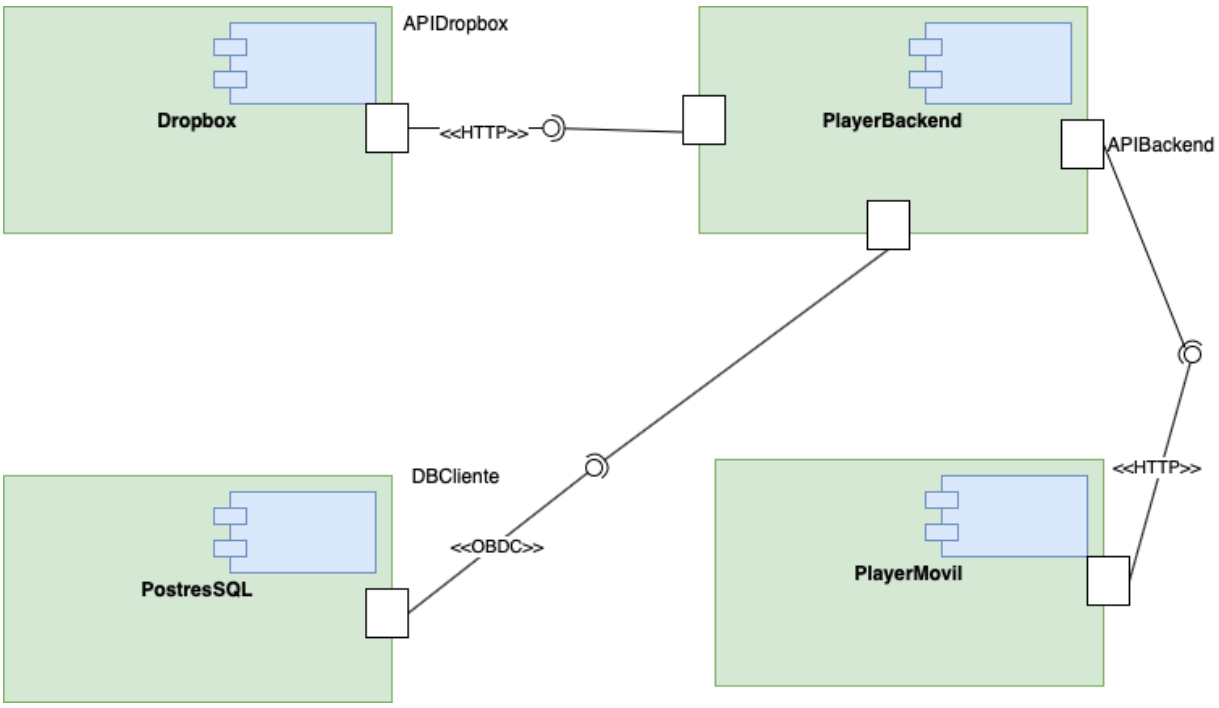


Nota adicional: Es necesario clarificar que para realizar todas y cada una de las actividades de la aplicación es necesario iniciar sesión, sin embargo, por motivos de estética y entendimiento no se han dibujado estas relaciones en el diagrama de casos de uso.

Esquema de datos (entidad-relación)



4.2.2 Diagrama de componentes y conectores



Las funciones de cada uno de los componentes y conectores serán explicadas a continuación:

PlayerMovil

Se trata de un componente de tipo proceso

Es el encargado de gestionar la cada presentación, es decir, la capa con la que interactúa el usuario final

Entre sus responsabilidades están la navegación por las diferentes pantallas, la lógica de botones, y la conexión con backend mediante el protocolo HTTP para añadir, eliminar o actualizar información en el sistema

PlayerBackend

Se trata de un componente tipo proceso.

Es el encargado de gestionar la lógica de negocio del sistema, su tarea es recibir toda la información de los usuarios y las partidas, procesarla y almacenarla de forma correcta.

PostgreSQL

Se trata de un componente de tipo almacenamiento

Su tarea es almacenar la información de las partidas y torneos y usuarios. Tambien debe gestionar toda la dinámica de una partida en vivo.

Dropbox

Se trata de un componente de tipo almacenamiento

Permite persistir las imágenes de perfil de cada usuario.

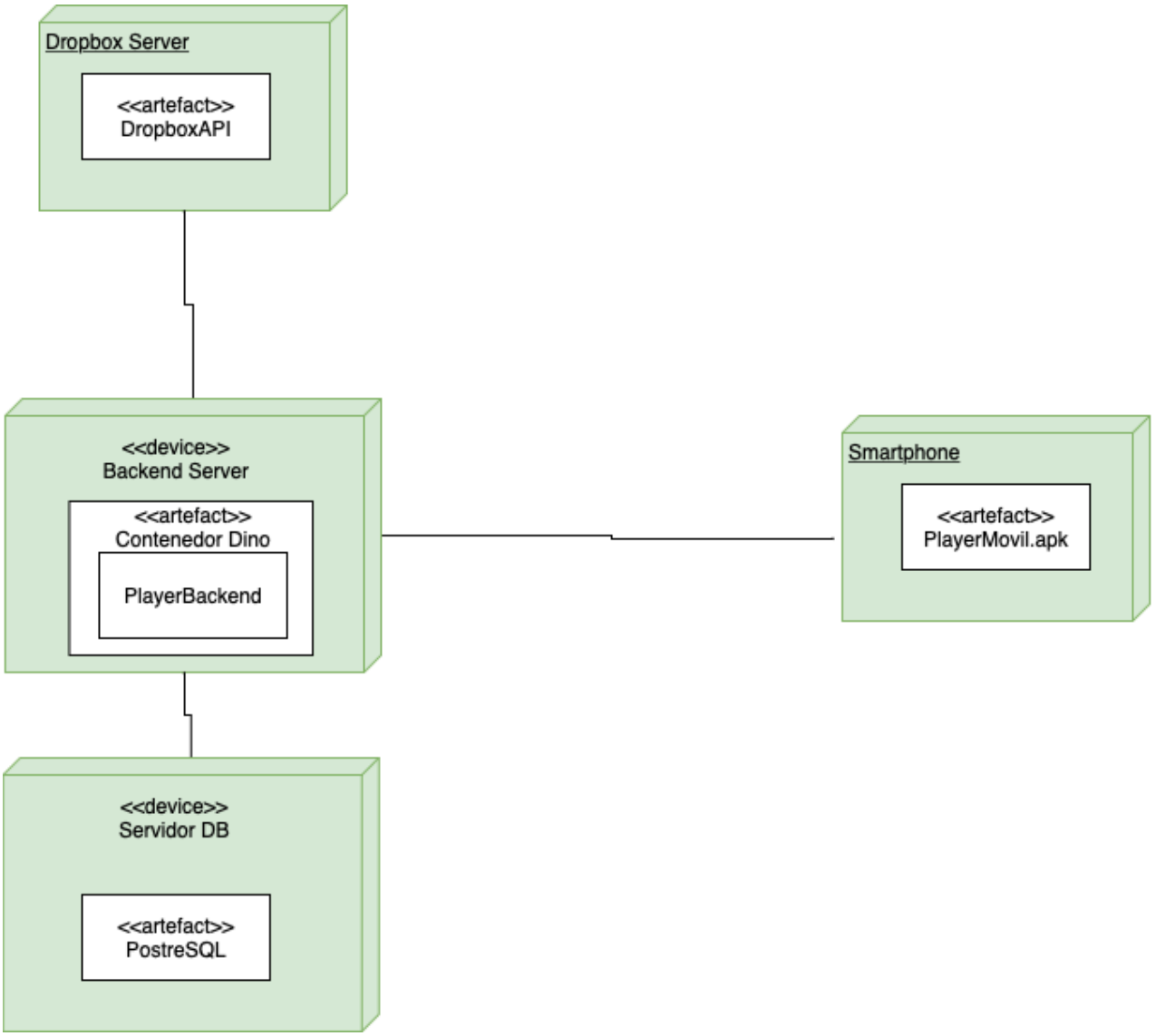
APIBackend

Es un puerto perteneciente a “PlayerBackend” el cual ofrece una API para la comunicación con “PlayerMovil”.

APIDropbox

Es un puerto perteneciente al componente “Dropbox” el cual logra el acceso de “PlayerBackend” a las fotos de perfil de los usuarios.

4.2.3 Diagrama de despliegue



Nodo Servidor Backend

El proyecto Django PlayerBackend se encontrará desplegado en el servicio de Heroku, encapsulado mediante un contenedor de tipo Dyno

Nodo Smartphone

Se trata de la APK, para desplegar en dispositivos Android con versión igual o superior a Android 5.0.

Servidor DB

Es la base de datos del proyecto, esta se despliega sobre el servicio encargado de almacenamiento de bases de datos perteneciente a Heroku

4.2.4 Prototipo de GUI

Será adjuntado junto con este documento en el formato de axure (aragote.rp). Se ha procedido a actuar de esta manera debido al gran tamaño y complejidad del diseño, asi como de sus interacciones táctiles (simuladas con ratón) imposibles de apreciar insertadas en un documento

5. Memoria del proyecto

La realización de este proyecto ha consistido principalmente en el desarrollo de un prototipo de GUI. Debido a que no se iba a realizar una implementación de la aplicación como tal, se ha realizado un prototipo avanzado que incluye la dinámica de muchas de las funcionalidades que tendrá el juego, las cuales serán nombradas seguidamente. Sin embargo ha habido otras, como toda la lógica a la hora de crear un torneo, que no han sido desarrolladas en su totalidad debido a la complejidad que estas implican.

De primera instancia se ha decido emplear **Axure RP 10** para el diseño de este prototipo, había alternativas interesantes como puede ser el famoso Adobe XD, sin embargo debido a los conocimientos previos sobre el funcionamiento de Axure y *la potencia* que este ofrece, se decidió que sería finalmente la plataforma escogida.

La aplicación prototipada se llama Aragote, viendo la popularidad del Guiñote en Aragón, así como variantes suyas (La Brisca) en el resto de España, se quiere implementar dicho juego con una interfaz moderna y actual que permita practicar este pasatiempo de forma Online, tanto con amigos como con desconocidos.

Para poder jugar una partida el jugador debe registrarse introduciendo sus datos personales, una vez registrado, únicamente tendrá que iniciar sesión con su usuario y contraseña.

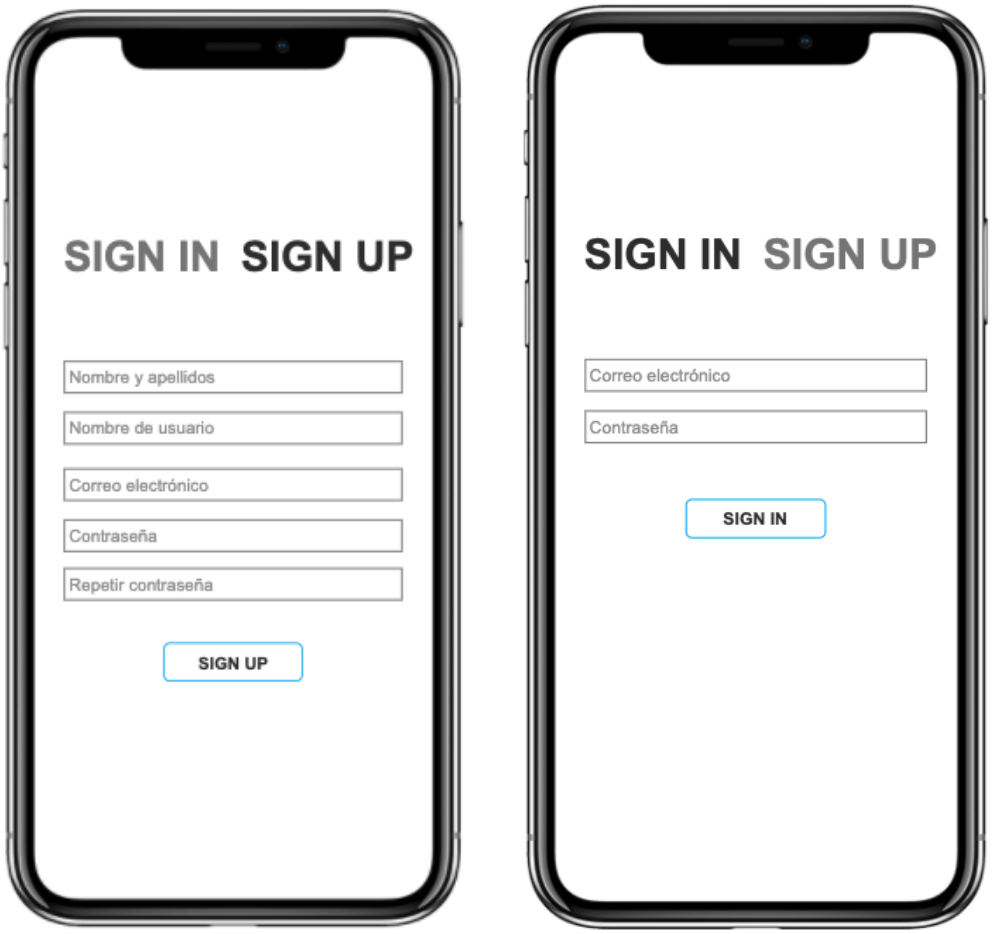


Imagen de izquierda, pantalla de registrarse

Imagen de derecha, pantalla de inicio sesión

Es en el menú principal donde este podrá comenzar a jugar, si directamente el usuario presiona el botón de “jugar partida”, el sistema le introducirá en una sala con otros tres jugadores de un nivel similar (explicados más adelante).
Sin embargo, el jugador puede escoger jugar con amigos (hechos previamente) pulsando el rectángulo detrás de su foto de perfil. Si éste acepta, les introducirá en una sala juntos, ambos formando equipo.

Imagen de izquierda, pantalla “home”

Imagen de derecha, pantalla buscando jugadores

Con cada partida ganada, entre otras cosas que se decidirán más adelante, el sistema otorgará puntos al jugador, que tras llenar a un cupo, subirá de nivel.

La dinámica de la partida seguirá las reglas tradicionales del guiñote aragonés. Cuando se inicie una de estas aparecerá el icono de todos los jugadores comprendiendo fácilmente quién es tu pareja y quien tu contrincante.

El turno de partida se indicará mediante una ficha de juego que se moverá a medida que cambian dichos turnos.

Durante la partida, al igual que en la vida real, se podrán consultar todas las bazas ganadas y la última baza ganada de tus contrincantes.

Imagen de izquierda, partida en la que tiene el turno "paulaPoker84"

Imagen de derecha, total de bazas ganadas por el equipo del usuario

En cada una de estas partidas, si uno de los participantes ha de marcharse, podrá pedir pausa, y si los demás están de acuerdo la partida quedará en pausa para ser continuada más adelante. Esto es una gran mejora respecto al juego tradicional, ya que para facilidad de todos, se introducen así las **partidas asíncronas.**

Imagen de izquierda, mensaje al pedir pausa

Imagen de derecha, mensaje al recibir petición de pausa

Imagen de izquierda, pantalla de pausa aceptada

Imagen de derecha, pantalla de pausa declinada

Además de poder realizar partidas simples, se ha introducido también el concepto de **torneo.** En el que se buscarán distintos participantes y de ellos se obtendrá un ganador. El torneo puede ser organizado por uno mismo, ser invitado o unirse a uno aleatoriamente.

Imagen de izquierda, pantalla de creación de torneo

Imagen de derecha, pantalla de búsqueda de participantes de torneo

Cada usuario, tiene un **perfil** en el que aparece su foto, nombre y nivel. El usuario puede modificar cuando quiera sus credenciales, así como el aspecto de la aplicación, esto es muy importante teniendo en cuenta la elevada edad de una gran parte de las personas que juegan actualmente.
Así entonces, los usuarios tienen la posibilidad de consultar los perfiles de otros usuarios, sean amigos o no, consultar su nivel, nombre, estadísticas…

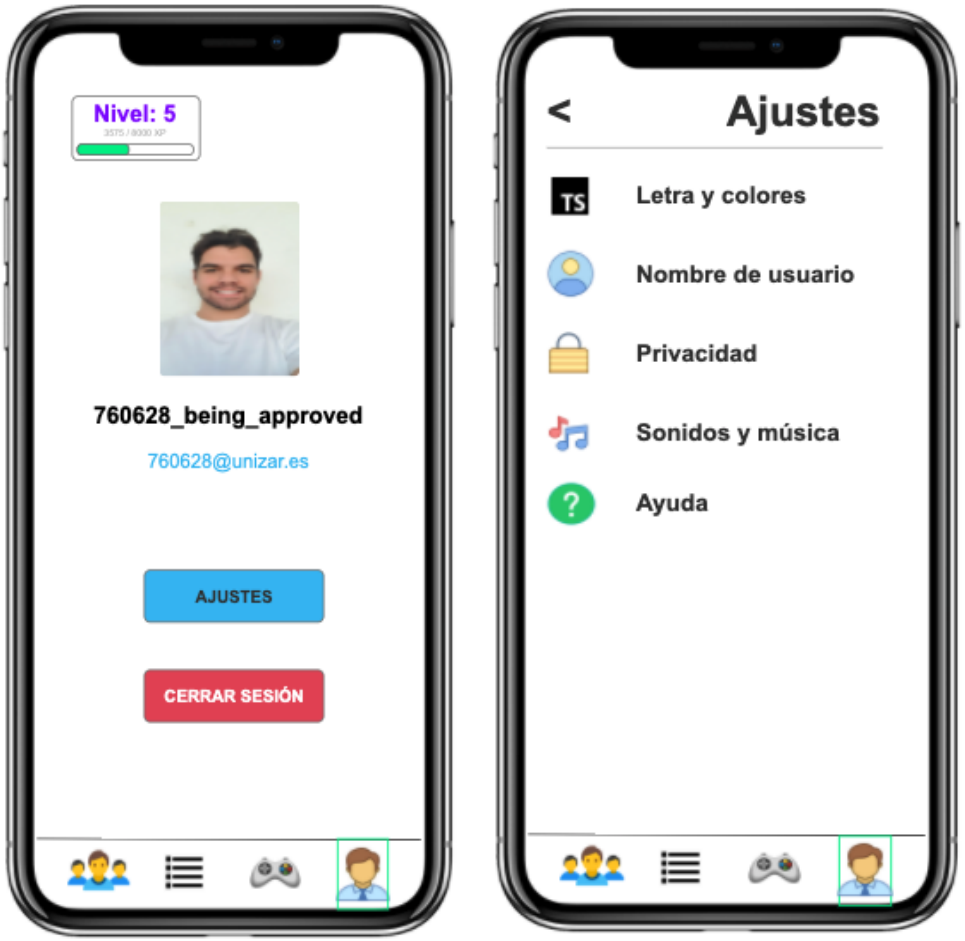


Imagen de izquierda, perfil de usuario
Imagen de derecha, ajustes disponibles

Cada jugador podrá revisar también sus estadísticas de cada partida o torneo, pudiendo ver quien ganó la partida, los demás participantes, las bazas que se ganaron etc.

Imagen de izquierda, registros de partidas

Imagen de derecha, pantalla de torneos

Información disponible al clicar en una partida

Tras ver todas las pantallas se observa una “**navbar**” que permite clicando en cada uno de los íconos desplazarse a lo largo de la aplicación. Esta tiene feedback ya que señala en todo momento en cual de los menús te encuentras

Imagen correspondiente a la navbar en el menú de perfil

Siguiendo con detalles técnicos del prototipo, se ha empleado la plantilla de un iPhone X, debido a que su diseño es de los más atractivos visualmente.

Para simular la aparición de elementos o similares se han empleado las acciones de Axure, en su mayoría de casos, al hacer tap en un elemento llevara a otra página en la que aparecerá la acción a mostrar, ya que el tener todos los elementos en la misma página y hacerlos aparecer o desaparecer puede generar confusión.

En ciertas pantallas se han escrito notas que facilitan la consulta y prueba del prototipo. Este ha sido probado en **Google Chrome**, ya que en otros navegadores como Safari genera acciones distintas a las planeadas.

Tras concluir el prototipo se obtiene un fichero RP, el cual puede ser exportado como imagen o mapa de bits.

6. Conclusiones

A pesar de únicamente haberse desarrollado un prototipo GUI, se han podido obtener conclusiones valiosas que ayudarán en la realización de futuros proyectos.

En primer lugar, a la hora de desarrollar un mockup como este siempre será conveniente pensar toda, o una gran parte de la lógica de aplicación antes, así como apoyarse con vistas, diagramas de despliegue etc.
El intentar diseñar una aplicación sin un diseño previo o una serie de pautas escritas en papel lo único que producirá es trabajar de manera ralentizada y obtener fallos o soluciones poco congruentes.

En lo que respecta a la comunicación con los miembros del equipo, es necesario mencionar que la presencialidad de las reuniones garantiza mayor entendimiento entre los participantes y el trabajar de forma más implicada. Si que es cierto que no siempre son necesarias estas, para dudas puntuales, o para esclarecer algún tema de corta discusión, el formato online genera en el participante más ganas de comunicarse con los miembros del equipo que si tuviera que desplazarse a un sitio alejado para únicamente resolver asuntos de menor importancia.

La no presencialidad a la hora de trabajar genera a su vez mayor heterogeneidad en la manera en la que se desarrolla el código. Es decir, se han visto comportamientos en una misma división de trabajadores, empleando la misma guía de estilos, en los que cada uno de estos suele desarrollar elementos de la manera en la que está más acostumbrada. Esto no debería ser así, ya que puede ocasionar incongruencias en los distintos módulos y causar fallos en la aplicación. Además, en caso de que posteriormente se integrarán nuevos miembros al equipo o se saliera uno de estos, sería más conveniente para la empresa mantener el mismo formato para facilitar el entendimiento del código.

Siguiendo con la organización del equipo, la decisión de que los mismos integrantes elijan la parte en la que van a trabajar (siempre que haya consenso) fomenta el trabajar agusto y la motivación individual. Para garantizar que se logran los objetivos pre impuestos es necesario que un miembro ajeno a una tarea realizada revise esta y genere informes, comentarios o dudas que tenga acerca de los elementos desarrollados, ya que si el encargado de realizar el estudio fuera el propio miembro de una división se corre el riesgo de que no sea muy crítico con el trabajo realizado, al tratarse del suyo propio.

Una buena observación obtenida durante los meses de labor, es que si se crea un equipo nuevo en los que los integrantes no se conocen y no existe un claro mando establecido, conviene tener alta precaución al escoger al director de proyecto. Si se da la situación de que se designa a este de forma casi aleatoria, únicamente por seguir los criterios de desarrollo es muy probable que su papel no aporte mucho al trabajo de equipo. A la hora de escoger el encargado, debe ser alguien con liderazgo, que en momentos de presión se venga arriba y que sea capaz de detectar cuando alguno de los miembros del equipo no está cumpliendo su labor, para llamar la atención y seguir con el correcto plan de trabajo.

En cuanto a detalles más técnicos, es de vital importancia establecer antes desarrollar ningún componente las configuraciones y bibliotecas que se van a usar. Si se emplean distintas entre los trabajadores es muy probable que la aplicación tenga en distintos sectores aspectos o comportamientos diferentes, o incluso que no compile. Además la sobrecarga de dichas bibliotecas, provocará seguro un mayor peso del sistema y un posible crecimiento de su lentitud. Sería poco eficiente emplear grandes módulos de los que apenas se va a usar su capacidad.

Asimismo, si la organización del proyecto consiste en una división de frontend y otra de backend, estos deberían comunicarse por adelantado antes de desarrollar módulos de gran importancia. Es comprensible, que al tratarse de aspectos tan “disjuntos” los trabajos de estos se paralicen y que ambos trabajen de forma independiente, en algunos tramos, a ritmos distintos, sin embargo es nuevamente muy importante la comunicación. Han surgido casos en los que la forma de entregar la API por parte de backend complica tremendamente la resolución necesitada en frontend, sacrificando aspectos de velocidad y rendimiento, únicamente por una mala comunicación. Por ello, es vital discutir los aspectos relativos al tratamiento de datos, a la forma en los que estos se van a obtener y al uso que se les dará a lo largo del desarrollo de la aplicación.

En último lugar se terminarán las conclusiones con dos premisas valiosas. No es necesario empezar a trabajar de manera acelerada, aporreando teclas y programando como si fuera una escena de “hacking hollywoodiense”. Plantear diseños previos, razonamientos pre trabajo y establecer con detalle cómo se realizaran las cosas, garantiza un resultado de mayor calidad y en menos tiempo.
Finalmente, el contacto y comunicación con los miembros del equipo es un activo que hay que cuidar. Comprender y ayudar a tus compañeros, generar un ambiente amable y profesional en el que exista una empatía en todas las direcciones, estimulará el afán individual y grupal de aportar al proyecto, sacar el plan adelante y lograr un trabajo de calidad del que todos sientan orgullo.

Glosario

Anexo II. Horas empleadas por el equipo de desarrollo

	Formación	Front-end	Back-end	Reuniones	Memoria	TOTAL*
Jose	20h	76h	-		15h	**111h**
Óscar	20h	100h	-		6h	**126h**
Josh	8h	89h	-		9h	**106h**
Sterling	15h	10h	80h		2h	**107h**
Steve	11h	59h	27h		7h	**104h**
John	7h	25h	78h		5h	**116h**

* Las horas totales no incluyen el periodo de formación en las distintas tecnologías.