2023
2024

Plan de gestión, análisis, diseño y memoria del proyecto CasinoLotus

PROYECTO SOFTWARE

GRUPO 08 – ROBERTA WILLIAMS

779035 - Bernad Ferrando, Lizer

816906 – Morales Rosa, Lucía

839841 – Moncasi Gosá, Carlota

839922 – López Ansón, Jesús

843350 - Terés Pueyo, Pablo

844419 - García Ortega, Marcos

851769 - Redondo Zamora, Aroa

852698 - Pavón Calcerrada, Claudia

Organización en GitHub: https://github.com/UNIZAR-30226-2024-08

Versión: v1.0.1

CONTENIDO

1 INTRODUCCIÓN

El proyecto se centra en el desarrollo de un casino virtual que ofrecerá la experiencia de jugar tanto al BlackJack como al Póker Texas Hold'em. Esta plataforma permitirá a los usuarios participar en partidas públicas o privadas, brindándoles la oportunidad de jugar con amigos si así lo desean. Además, los usuarios podrán personalizar sus cuentas mediante elementos adquiribles con el dinero obtenido de las partidas ganadas.

El propósito principal del proyecto es proporcionar una plataforma de entretenimiento virtual que ofrezca una experiencia de juego de casino realista y socialmente interactiva. Al facilitar la personalización de las cuentas de usuario y la posibilidad de jugar con amigos, se busca crear un ambiente atractivo y envolvente para los jugadores.

El alcance del proyecto incluye el desarrollo completo de la plataforma de casino virtual, desde la implementación de los juegos de BlackJack y Póker Texas Hold'em hasta la integración de funcionalidades sociales que permitan a los usuarios jugar con amigos, personalizar sus cuentas y pausar las partidas para retomarlas en otro momento. Se considerará la posibilidad de ampliar la oferta de juegos en etapas posteriores del desarrollo.

Los objetivos principales son las funcionalidades que se han comentado anteriormente, como la implementación de ambos juegos, la posibilidad de personalizar las cartas y la foto de perfil a través de compras en la tienda de la aplicación y la posibilidad de pausar partidas para retomarlas más adelante, siempre que todos los participantes estén conectados. El sistema que se realizará será multiplataforma, por lo que permitirá el acceso no simultáneo de un usuario en distintas plataformas.

Al final del proyecto se entregará al cliente una dirección web donde encontrará la página web funcional del proyecto. Además, se les entregará un ejecutable para el uso de la aplicación desde el escritorio y una forma de acceso segura a los servicios de Azure donde se habrá desplegado la aplicación. También se hará entrega del plan de gestión, análisis y diseño añadido a la memoria del proyecto, un documento de especificación de requisitos, un documento con la guía de la arquitectura empleada, una documentación de la API y una guía de despliegue.

Los principales hitos planteados para el proyecto son la creación de una API que permita unir los servicios proporcionados por el backend con las peticiones realizadas por los frontend que se pretende tener realizado para el 17 de marzo. Después se va a tratar de tener lo necesario para el registro y autenticación de usuarios para el domingo 31 de marzo. Tras esto se data un corto plazo de tiempo para finalizar las implementaciones del desarrollo del Blackjack, de forma que para el 7 de abril el juego sea funcional y se pueda mostrar en la primera entrega del proyecto. Después se realizará la implementación del segundo juego con fecha límite el 21 de abril y se terminarán de implementar las demás funcionalidades menores para la fecha de entrega final del proyecto. Además, se pretende revisar el diagrama de Gantt, mostrado en la sección Calendario del proyecto, división del trabajo, coordinación, comunicaciones, monitorización y seguimiento, cada tres semanas para reajustar lo necesario.

Este documento contendrá la organización del equipo encargado del proyecto, un plan de gestión que explica cómo se realizarán las tareas necesarias para acabar el proyecto, y

cómo se organizan los recursos empleados en ellas. También se detallan los requisitos del sistema y el diseño principal de este. Además, se muestra la evolución del proyecto, de inicio a fin. Y por último las conclusiones sacadas al realizar el trabajo, seguido del glosario y varios anexos que aportarán información adicional.

2 ORGANIZACIÓN DEL PROYECTO

El equipo destinado a la realización del proyecto está formado por ocho integrantes. Estos integrantes han sido asignados a diferentes sectores por el director del proyecto, teniendo en cuenta las preferencias y fortalezas de cada miembro. Como resultado, se ha obtenido una distribución balanceada entre las distintas tareas principales del trabajo, destinando a dos integrantes a realizar el trabajo del backend, y seis para la realización de los frontend, dividiéndose tres para el desarrollo del frontend para la aplicación web y otros tres para el frontend de la aplicación de escritorio.

Es necesario puntualizar que cuando se termine el trabajo del backend, que se prevé que será el primero en finalizar, sus integrantes serán asignados a otras tareas.

Además, se ha decidido que el director del proyecto comience trabajando en el backend, pero dispondrá de flexibilidad para cambiarse a otras tareas si se requiere de su ayuda. De esta forma, aparte de ofrecer soporte de ser necesario, se asegurará la consistencia entre las diferentes secciones.

En la Tabla 1 se detalla el reparto de miembros entre las tareas mencionadas anteriormente.

Miembro del equipo	**Tarea**
López Ansón, Jesús	Director del proyecto
Bernad Ferrando, Lizer	Desarrollador backend
Moncasi Gosá, Carlota	Desarrolladora backend
Morales Rosa, Lucía	Desarrolladora del frontend para web
Redondo Zamora, Aroa	Desarrolladora del frontend para web
García Ortega, Marcos	Desarrollador del frontend para web
Terés Pueyo, Pablo	Desarrollador del frontend para escritorio
Pavón Calcerrada, Claudia	Desarrolladora del frontend para escritorio

Tabla 1: Reparto inicial de tareas de los integrantes del equipo

Por otro lado, se han asignado diferentes roles a algunos miembros del equipo. Estos miembros serán los encargados principales de llevar a cabo las distintas tareas detalladas en la Tabla 2.

Rol asignado	**Explicación del rol**	**Miembro del equipo**
Contacto con los clientes	Responsable de la comunicación con los clientes.	López Ansón, Jesús
Responsable de GitHub	Encargado de solucionar problemas relacionados con la herramienta de GitHub	López Ansón, Jesús Bernad Ferrando, Lizer
Aprobador de documentos	Responsable de la aprobación final de documentos que se consideren terminados.	López Ansón, Jesús

Construcción del software.	Responsable de la correcta documentación del software.	Bernad Ferrando, Lizer
Tomador de notas	Responsable de tomar notas en las reuniones de los miembros del equipo o con los clientes.	Moncasi Gosá, Carlota
Gestor de esfuerzos	Responsable de apuntar las horas semanales trabajadas por cada miembro del equipo.	Morales Rosa, Lucía
Responsable backend	Responsable de la asignación de tareas del backend.	Bernad Ferrando, Lizer
Responsable frontend web	Responsable de la asignación de tareas del frontend para web.	Morales Rosa, Lucía
Responsable frontend escritorio	Responsable de la asignación de tareas del frontend para escritorio.	Terés Pueyo, Pablo
Aseguramiento del producto	Responsable de la revisión y correcta gestión de las pruebas.	García Ortega, Marcos
Calendario del proyecto	Responsable de la gestión del proyecto y el correcto cumplimiento de las tareas	López Ansón, Jesús
Gestor de documentación	Responsable de realizar un seguimiento de los documentos y asegurar que están actualizados.	Redondo Zamora, Aroa
Diseño gráfico	Responsable de los recursos gráficos que empleará la aplicación	Pavón Calcerrada, Claudia

Tabla 2: Roles y responsabilidades de los integrantes del equipo

3 PLAN DE GESTIÓN DEL PROYECTO

En esta sección se va a detallar cómo se van a llevar a cabo las distintas tareas que se deben realizar durante el proyecto y los recursos que se emplearán.

3.1 Inicio del proyecto

Se ha considerado que para el proyecto planteado solo será necesario el uso de los servicios y recursos proporcionados por Microsoft Azure para alojar la base de datos y lo necesario para realizar el despliegue del backend y del frontend de web.

Por otro lado, en cuanto a las tecnologías elegidas se han estudiado diversas opciones ya que, como ningún integrante del grupo había realizado con anterioridad un proyecto similar, los conocimientos sobre las diferentes posibilidades a escoger eran escasas.

Tras realizar un estudio de las diferentes propuestas encontradas, que se comenta con mayor detalle en la sección 4.2, se ha decidido emplear un *stack* MyEAN para el backend y el desarrollo del frontend de web y Godot para el desarrollo del frontend de escritorio.

Para poder llevar a cabo el proyecto ha sido fundamental la formación de los miembros del equipo en sus respectivas áreas de trabajo, especialmente teniendo en cuenta que la mayoría de los integrantes no han empleado ninguna de las tecnologías mencionadas con anterioridad. Por ello, se ha decidido hacer uso de las primeras semanas para formar a los distintos grupos de desarrollo en las tecnologías seleccionadas. Esta formación se ha basado principalmente en la visualización e implementación de distintos tutoriales que se han encontrado en YouTube y se han considerado suficientemente completos como para aprender lo esencial de estas tecnologías.

En concreto, cabe la pena mencionar los tutoriales más destacados que se han empleado para la formación de los miembros del equipo en las distintas disciplinas. Para el frontend de web se ha utilizado el video de ANGULAR desde cero de Sergie Code. Para entender como relacionar el código del frontend de web con el backend a través de la API han sido especialmente útiles el video Angular Mysql CRUD de Fazt y el curso Curso de Node-JS de midulive. Por otro lado, para el desarrollo de escritorio se ha empleado directamente la documentación técnica de Godot.

Estos, como se ha comentado, son solo algunos de los tutoriales que se han usado para la formación y que se han considerado más relevantes y de mayor aporte.

3.2 Control de configuraciones, construcción del software y aseguramiento del producto

Para asegurar la corrección en los distintos aspectos se han designado a distintos responsables que se han detallado en la Tabla 2. Concretamente, se ha decidido que haya dos responsables del control de configuraciones que serán Jesús López y Lizer Bernad, el responsable de la construcción del software será Lizer Bernad y el del aseguramiento del producto será Marcos García.

En cuanto a la forma de nombrar los ficheros se ha decidido utilizar *camelCase* y prescindir de los espacios y tildes. Se empleará el guion bajo para separar el nombre de otros

elementos de interés y característicos del documento como lo podría ser la versión del fichero y los nombres deberán ser cortos y descriptivos del contenido del archivo. En cuanto al formato del código se ha decidido emplear la extensión "Prettier" de Visual Studio Code ya que la mayoría del equipo emplea ese entorno y facilita en gran medida la maquetación del código.

Respecto al control de versiones se ha empleará la herramienta GitHub. Para mantener el orden y la coherencia se separará el contenido en cuatro repositorios "back-end", "front-Escritorio", "front-Web" y "documentacion". Todos los miembros del equipo tienen permisos de administrador para todos los repositorios, aunque los responsables de GitHub serán los encargados de realizar determinadas tareas como los *merge* entre ramas.

Para la gestión de incidencias se hará uso de la herramienta GitHub Issues de forma que los miembros del equipo puedan asignar tareas a otros miembros o a sí mismos y se pueda llevar un seguimiento sencillo de las distintas tareas que está realizando cada uno. Las incidencias tendrán distintos estados: sin estado, por hacer, en proceso y hecho. Este estado podrá ser modificado por cualquiera de los miembros involucrados en la incidencia, es decir, tanto el creador como los asignados para resolverla.

Además, se crearán distintos *milestones* o hitos para indicar las fechas límites de distintas tareas y se hará uso de las *labels* o etiquetas para añadir información a cada tarea, permitiendo también una mejor clasificación de estas. Para poder concretar un poco más los tipos de errores que aparecerán a la hora de desarrollar el código, se han creado unas subetiquetas de la etiqueta predeterminada "*bug*" con los que se le podrá asignar distintos niveles de error a los problemas encontrados.

A la hora de realizar el despliegue, se ha decidido que se empleará el servicio Microsoft Azure para desplegar tanto el backend como la aplicación de angular, permitiendo así el acceso a la página web de forma remota con el enlace proporcionado por Azure. Por otro lado, el despliegue de la aplicación de escritorio se generará desde Godot un ejecutable que permita su instalación.

En cuanto a cómo realizar cambios en la documentación y código fuente, se empleará el *workflow* de trabajo GitHubFlow cuyos puntos principales son que se creará una rama para cada nueva funcionalidad a implementar o problema a resolver y se tratará de dejar en la rama principal el código lo más correcto y funcional posible. Para realizar los *merge* entre ramas, se deberá indicar con una incidencia o *pull-request* a los responsables de GitHub que la rama se ha finalizado y está lista para ser juntada con el trabajo principal. Los *commits* no tendrán que ser aprobados por ningún miembro del equipo, pero tendrán que tener un título descriptivo y una descripción que detalle los cambios realizados.

Se crearán scripts de construcción de software si fuera necesario. En el caso del software de escritorio, el ejecutable puede ser creado fácilmente para todas las plataformas, por lo que no será necesario utilizar ningún script. Además, se tiene el objetivo de integrar el proyecto una vez cada dos semanas.

Para llevar a cabo una buena documentación del proyecto, se tiene pensado desarrollar distintos documentos que serán entregado al cliente junto al acceso al código fuente a través de GitHub, el acceso controlado a los sistemas desplegados en Azure y los ejecutables de escritorio para Linux y Windows. Este conjunto de documentos se tiene

previsto que esté formado por un documento de especificación de requisitos, un documento con la guía de la arquitectura empleada, una documentación de API y una guía de despliegue. Por otro lado, en cuanto al diseño de la interfaz gráfica de usuario se tiene como prioridad alcanzar una accesibilidad óptima para permitir que los usuarios puedan interactuar de manera efectiva con la aplicación y acceder a toda su funcionalidad de manera inclusiva y equitativa. Para ello se va a hacer uso de algunas de las reglas que provee el estándar WCAG 2.0[a]. Además, para mejorar la experiencia de los usuarios se tratará de seguir con la mayor rigurosidad las 8 reglas de oro de Schneiderman[b] y los Principios de la Gestalt[c].

En cuanto a la corrección y aseguramiento de la calidad del proyecto, se han tomado diferentes medidas. Por ejemplo, para asegurar la corrección del código y documentación, se asignarán personas diferentes a la redacción y revisión o, como mínimo, en caso de ser revisado por la misma persona, la revisión se realizará unos días después de la redacción. Además, el director de proyecto deberá realizar una última revisión de todos los documentos que se consideren terminados para dar su aprobación, por lo que se contará con una revisión extra. Respecto a la comprobación de la corrección del código, se ha decidido la creación de test automáticos de forma constante para detectar las deficiencias lo antes posible y poder ser corregidas sin grandes complicaciones.

3.3 Calendario del proyecto, división del trabajo, coordinación, comunicaciones, monitorización y seguimiento

El responsable del calendario del proyecto es Jesús López, que por lo tanto también se encarga de la creación de los *milestones* en GitHub. Además, hay un responsable por cada grupo de trabajo principal, Lizer Bernad para backend, Lucía Morales para frontend web y Pablo Terés para frontend escritorio.

En el diagrama de Gantt mostrado en la Ilustración 1, se representa la primera versión de organización planteada para el desarrollo del proyecto. Este diagrama se revisará cada tres semanas y se modificará en caso de ser necesario para ajustar los tiempos previstos.

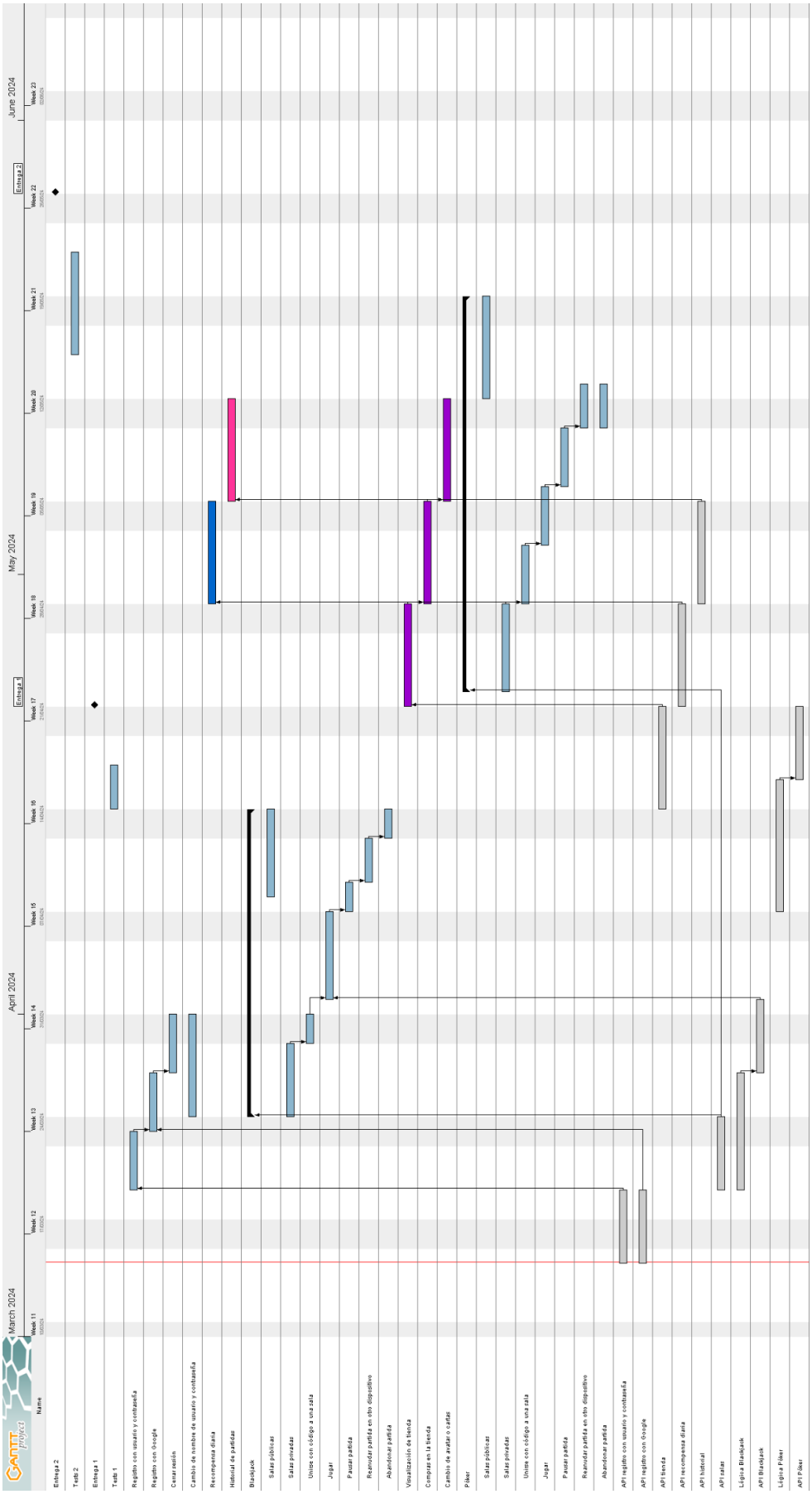


Ilustración 1: Primera versión del diagrama de Gantt con la planificación planteada

Para realizar las comunicaciones internas entre los miembros del equipo se hará uso de la herramienta GitHub Issues para aquellas interacciones que hagan referencia a secciones concretas del código o la documentación y un grupo de Whatsapp común por el que se discutirán diferentes ideas y se concretarán horas para las reuniones de equipo necesarias. En estas reuniones habrá un responsable que tomará nota de las decisiones tomadas y después estas ideas se recogerán en un documento compartido para que todos los integrantes tengan acceso a él y puedan consultarlo si es preciso.

Las tareas diarias a realizar por el equipo serán asignadas de forma semanal por los responsables de cada sección junto con la colaboración del director de proyecto, de forma que siempre habrá alguien que esté al tanto de todo lo que se está desarrollando.

Teniendo esto en cuenta, la figura del director también servirá para monitorizar el progreso y el estado general del proyecto y como la asignación de tareas se realizará de forma semanal, el control del trabajo se realizará de forma lo suficientemente constante como para detectar problemas en el avance o de rendimiento. En caso de que esto ocurra, el director hablará con el responsable del módulo responsable del retraso y este hablará a su vez con la persona causante del retraso para averiguar el motivo. Tras esto, se le notificará nuevamente al director, que será el responsable de decidir si es necesario tomar medidas como destinar a más miembros a trabajar en el módulo afectado o si ha sido un problema puntual que se podrá subsanar de forma eficaz.

En el caso de que haya disputas entre algunos miembros del equipo se realizará una votación general de los involucrados para tratar de llegar a una conclusión mayoritaria y si el problema no se resuelve, entonces el director tomará partido y tras escuchar a todas las partes tomará una decisión, teniendo en cuenta cómo afectará lo decidido al resto del proyecto.

4 ANÁLISIS Y DISEÑO DEL SISTEMA

4.1 Análisis de requisitos

Para evitar ambigüedades y definir los conceptos de forma correcta se han detallado los grupos de interés, un diccionario de datos y un glosario de términos en el Anexo III. Complemento al análisis de requisitos.

En la Tabla 3 se detallan los requisitos funcionales de los diferentes *stakeholders*.

Requisitos Funcionales de los Usuarios No Autenticados	
RFUNA-1.	El sistema permitirá autenticarse con su nombre de usuario y contraseña.
RFUNA-2.	El sistema permitirá registrarse con una cuenta de Google no registrada en el sistema.
RFUNA-3.	El sistema permitirá salir de la aplicación.
Requisitos Funcionales de los Usuarios No Autenticados	
RFUA-1.	El sistema permitirá cerrar sesión.
RFUA-2.	El sistema permitirá salir de la aplicación.
RFUA-3.	
a.	El sistema permitirá crear una sala privada[6] de Blackjack[1].
b.	El sistema permitirá crear una sala privada[6] de Póker[2].
c.	El sistema permitirá crear una sala pública[7] de Blackjack[1].
d.	El sistema permitirá crear una sala pública[7] de Póker[2].
RFUA-4.	El sistema permitirá unirse como jugador[12] a una sala[3] con un código de invitación.
RFUA-5.	
a.	El sistema permitirá solicitar la reanudación de una sala iniciada[5] previamente pausada.
b.	El sistema permitirá abandonar una sala iniciada[5] previamente pausada.
RFUA-6.	
a.	El sistema permitirá cambiar el nombre de usuario si no existe otro usuario con el nuevo nombre de usuario.
b.	El sistema permitirá cambiar la apariencia de cartas si la nueva está desbloqueada.
c.	El sistema permitirá cambiar el avatar si el nuevo está desbloqueado.
RFUA-7.	El sistema permitirá consultar el historial de partidas[8].
RFUA-8.	El sistema permitirá la compra de personalizables[15].
Requisitos funcionales de los jugadores	

RFJ-1.	
a.	El sistema permitirá iniciar una sala de Blackjack[1] pública[7] creada[4] que contenga de 1 hasta 7 jugadores[12].
b.	El sistema permitirá iniciar una sala de Blackjack[1] privada[6] creada[4] que contenga de 1 hasta 7 jugadores[12].
c.	El sistema permitirá iniciar una sala de Póker[2] pública[7] creada[4] que contenga de 1 hasta 10 jugadores[12].
d.	El sistema permitirá iniciar una sala de Póker[2] privada[6] creada[4] que contenga de 2 hasta 10 jugadores[12].
RFJ-2.	El sistema permitirá abandonar la sala creada[4].
Requisitos Funcionales de los Jugadores de Blackjack	
RFJB-1.	
a.	El sistema permitirá pausar la sala iniciada[5].
b.	El sistema permitirá abandonar la sala iniciada[5].
RFJB-2.	
a.	El sistema permitirá pedir carta durante su turno si la mano[9] no supera 21 puntos.
b.	El sistema permitirá plantarse durante su turno si la mano[9] no supera 21 puntos.
RFJB-3.	El sistema permitirá apostar al inicio de una partida[8] una cantidad entre 1 y el saldo.
Requisitos Funcionales de los Jugadores de Póker	
RFJP-1.	
a.	El sistema permitirá pausar la sala iniciada[5].
b.	El sistema permitirá abandonar la sala iniciada[5].
RFJP-2.	
a.	El sistema permitirá igualar una apuesta en su turno a una cantidad entre 1 y el saldo cumpliendo las reglas de apuestas del Póker[2].
b.	El sistema permitirá subir una apuesta en su turno a una cantidad entre 1 y el saldo cumpliendo las reglas de apuestas del Póker[2].
c.	El sistema permitirá retirarse en su turno.

Tabla 3: Requisitos funcionales

También se han especificado los requisitos no funcionales en la Tabla 4.

Requisitos No Funcionales	
RNF-1.	El sistema permitirá partidas síncronas[10].
RNF-2.	El sistema permitirá autenticarse en distintos dispositivos de forma no simultánea.
RNF-3.	

a.	El sistema recompensará a los ganadores de una partida[8] el monto de moneda virtual[14] respetando las reglas del juego[13].
b.	El sistema recompensará a los jugadores[12] con 500 monedas virtuales[14] al autenticarse no más de una vez en las últimas 24 horas.
RNF-4.	El sistema deberá mantener la partida[8] si uno o más jugadores[12] pierden la conexión hasta que llegue el turno de alguno de esos jugadores[12].
RNF-5.	El sistema pausará la partida[8] si un jugador[12] llega a su turno sin conexión.
RNF-6.	El sistema no permitirá la compra de un personalizable[15] si el usuario no dispone un número de monedas virtuales[14] igual o superior al precio del personalizable[15]

Tabla 4: Requisitos no funcionales

Además, se han detectado algunas restricciones que el sistema debe cumplir y se detallan en la Tabla 5.

Restricciones	
1	
a.	El sistema deberá desplegarse en un dispositivo con conexión a internet a través de un navegador web.
b.	El sistema deberá desplegarse en un dispositivo con conexión a internet a través de una aplicación de escritorio.

Tabla 5: Restricciones

Además de las tablas de requisitos y restricciones se ha decidido realizar una formalización estructurada de los mismos, de forma que los posteriores análisis que se realicen en base a ellos resulten mucho más sencillos de llevar a cabo. Esta formalización de los requisitos se detalla en el Anexo IV: Formalización de requisitos.

4.2 Diseño del sistema

Para la realización del proyecto se ha decidido emplear el Modelo Vista Controlador en una arquitectura de tres *tiers*, para lo que se han llevado a cabo varios diagramas arquitecturales para facilitar las tareas posteriores de implementación.

Se ha comenzado diseñando un diagrama de casos de uso para los requisitos planteados inicialmente de forma que se puedan visualizar de forma más cómoda, tal y como se muestra en la Ilustración 2.

Paralelamente se ha creado la vista de módulos del backend, que se detalla en el Anexo V: Vista de módulos del backend.

Ilustración 2: Diagrama de casos de uso

En la Ilustración 3 se muestra el diagrama de componente y conector que se ha desarrollado para visualizar y comprender la estructura interna del sistema y como los componentes se relacionan entre sí.

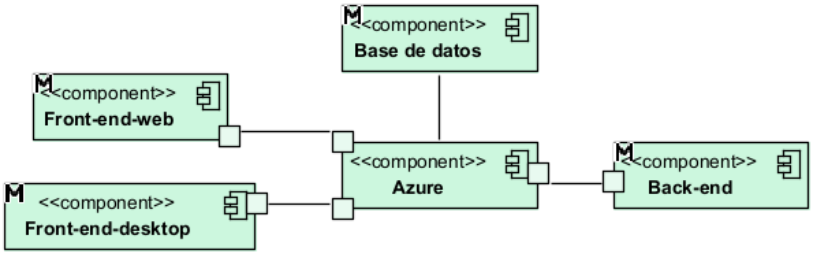


Ilustración 3: Vista componente y conector

También se ha diseñado un diagrama de despliegue que se muestra en Ilustración 4 para representar cómo se distribuyen los componentes del sistema en el hardware físico y en el entorno de ejecución.

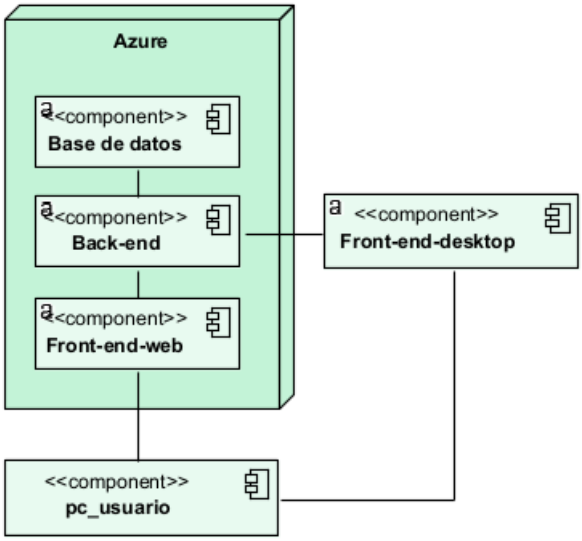


Ilustración 4: Vista de despliegue

Como ya se había comentado con anterioridad, se ha optado por emplear Angular para el desarrollo del frontend de web, Godot para el frontend de escritorio y Node JS junto a Express para realizar el backend. En cuanto a la base de datos se ha decidido emplear el gestor de bases de datos relacional MySQL.

Sin embargo, hubo diferentes opciones a valorar a la hora de elegir las tecnologías que se iban a emplear para el desarrollo del proyecto.

Entre las opciones valoradas para el desarrollo del frontend para web se encuentran REACT y Angular. Tras informarse debidamente sobre ambas tecnologías y realizar algunas pruebas sencillas, se decidió emplear Angular ya que se consideró que su curva de aprendizaje sería más amigable para principiantes que la de REACT.

Una vez decidida la tecnología principal para el desarrollo web se investigó sobre las posibles opciones para el backend y se observó que un *stack* muy recomendado en diversos portales era el *stack* MEAN, que emplea MongoDB como gestor de bases de

datos, Angular para el frontend y Node JS y Express para el backend y la gestión de las APIs.

Con esto en mente, se realizaron algunas pruebas simples con estas tecnologías y se decidió que Node JS y Express se emplearían en el desarrollo del backend, pero se optó por emplear MySQL como gestor de bases de datos ya que todos los miembros del equipo habíamos empleado con anterioridad bases de datos relacionales y resultaría más sencillo implementar la base de datos de esta forma en lugar de estudiar cómo realizar una base de datos no relacional. También se valoró usar PostgreSQL, pero se descartó ya que, a pesar de que ambas funcionarían correctamente teniendo en cuenta las necesidades del proyecto, se ha considerado que las consultas a realizar a la base de datos no serán demasiado complejas ni se actualizarán los datos de forma frecuente, por lo que MySQL es ligeramente más adecuada.

Por otro lado, para el desarrollo de escritorio se ha optado por emplear Godot, ya que, pese a que todos los miembros del equipo han empleado Android Studio con anterioridad, algunos de ellos habían empleado también Godot y consideraron que, en su experiencia, era más efectivo emplear esta tecnología. Además, ya contaban con un pequeño manejo del entorno por lo que se reduciría la necesidad de formación de estos integrantes.

En resumen, se ha optado por emplear un *stack* MyEAN para el desarrollo web y Godot para el desarrollo en escritorio.

En cuanto al lenguaje de programación, se ha elegido TypeScript para el desarrollo de backend y frontend de web. La razón de emplear este lenguaje es que es el que se emplea generalmente en Angular y, como los integrantes del grupo de backend posteriormente ayudarán en frontend, se ha decidido que también comiencen a usar TypeScript para familiarizarse con él desde el principio. Para el desarrollo del frontend de escritorio se ha optado por emplear GDScript ya que es el lenguaje que ambos miembros del equipo conocían con anterioridad.

5 MEMORIA DEL PROYECTO

En este capítulo se describirá cómo se ha llevado a cabo el proyecto hasta el momento, qué cambios se han hecho respecto a la versión inicial planteada en la sección 3.1 y los distintos imprevistos que han surgido.

5.1 Inicio del proyecto

A la hora de comenzar el proyecto, se dedicó la primera semana a la discusión de las tecnologías que se iban a emplear. Esta tarea fue algo más complicada de lo esperado ya que el profundo desconocimiento de todos los miembros del equipo dificultó ligeramente el consenso.

Una vez decidas las tecnologías se planificó un máximo de dos semanas para la formación de los miembros en los entornos y lenguajes asignados a la vez que se empezaba a crear el documento de especificación de requisitos. Esta estimación de dos semanas no fue del todo correcta, especialmente en el grupo destinado al desarrollo del frontend de web, que empleó alguna semana más para terminar su formación.

Como se ha mencionado anteriormente, simultáneamente a la formación se fue desarrollando una especificación de requisitos, a la que se le dedicó bastante tiempo puesto que se consideró que una correcta definición de los requisitos permitirá al equipo tener claras las bases del proyecto y evitará continuos problemas en el futuro.

5.2 Ejecución y control del proyecto

El reparto de trabajo entre miembros del equipo se repartió tal y como se detalla en la sección 2 y la comunicación interna se ha realizado principalmente vía WhatsApp. Esto pretende mejorarse próximamente de forma que se emplee la herramienta de GitHub Issues más eficientemente.

En cuanto al curso de trabajo, tras la formación del equipo que se menciona en el apartado anterior, el siguiente paso lógico fue el de comprobar el uso de los conocimientos adquiridos a través de los diversos tutoriales de Youtube y documentación de las distintas tecnologías usadas en este proyecto, como la documentación oficial de Godot o de Angular.

Además, como en este punto se poseen los conocimientos suficientes sobre las herramientas como para determinar qué es lo que se puede y no se puede hacer, y cuánto tiempo estimado se tarda en realizar cada tarea, se ha procedido a replantear los hitos. Esto permitirá distribuir mejor las tareas a realizar en el futuro, y por lo tanto permitirá a los miembros del equipo tener siempre una meta que cumplir. Esto también será útil para medir el progreso del proyecto de manera más objetiva y poder determinar de forma simple si se va cumpliendo con el plan en todos los sectores o es necesario tomar medidas.

Concretamente, en el desarrollo de frontend de web, los esfuerzos se han centrado en realizar tareas básicas para comprobar el correcto funcionamiento de la interfaz de usuario como la navegación entre pestañas o la creación de componentes padre/hijo y la gestión del paso de información entre ellos. En general, conocer cómo implementar la interfaz de forma efectiva en web, ya que es un medio más complicado que en aplicación

nativa de escritorio. Además, se ha conseguido realizar una prueba para el despliegue de la aplicación de Angular en Microsoft Azure.

Por otro lado, en cuanto al frontend de escritorio, se ha tratado de implementar una interfaz de usuario que funcione y que sea fácilmente reemplazable por los recursos que podamos encontrar en el futuro para que sea más accesible y atractiva para el usuario. Debido a que la aplicación permite una interacción más fácil a través de su interfaz gráfica, no se ha necesitado tanta formación como en el otro equipo de frontend, y ha permitido la búsqueda de estos recursos con los que cambiar los paneles y botones básicos por unos más acordes a la temática.

Además, como trabajo común entre los frontend, se han diseñado prototipos de las pantallas a implementar, de forma que ambas interfaces sean lo más similares posibles dentro de lo razonable. También se han seleccionado recursos gráficos que se van a compartir para ambos frontend como la baraja de cartas a usar.

Por último, el equipo de backend se ha centrado en terminar y revisar el modelo entidad – relación y el modelo relacional para realizar la base de datos. También se ha actualizado el *Readme* y *.gitignore* del repositorio de GitHub de forma que solo se compartan los archivos necesarios y se ha comenzado a implementar la API para la conexión de los frontend con las funcionalidades ofrecidas por el backend.

En cuanto aspectos a terminar cuanto antes está finalizar la parte de backend de la API para que los frontend pueda conectarse y la vista de módulos (VO, DAO, Gestor) del sistema.

5.3 Cierre del proyecto

A completar.

6 CONCLUSIONES

A completar.

GLOSARIO

ANEXO I. PRESUPUESTO

A completar.

ANEXO II. RESULTADOS DE LA REVISIÓN INTERMEDIA

<<Entrega final solo>>

ANEXO III. COMPLEMENTO AL ANÁLISIS DE REQUISITOS

Descripción de los grupos de interés

Usuario no Autenticado[11]: Usuario que no se ha autenticado en el sistema.

Usuario Autenticado[11]: Usuario que se ha autenticado en el sistema con éxito.

Jugador[12]: Usuario que se encuentra en una partida[8] creada y forma parte del conjunto de jugadores de esta.

Jugador[12] de Blackjack: Usuario que se encuentra en una partida[8] iniciada de Blackjack[1] y forma parte del conjunto de jugadores[12] de la misma.

Jugador[12] de Póker: Usuario que se encuentra en una partida[8] iniciada de Póker y forma parte del conjunto de jugadores[12] de la misma.

Diccionario de datos

Usuario Registrado: Los datos de un usuario registrado incluyen un nombre de usuario, un correo asociado a una cuenta de Google, un avatar, una apariencia de cartas, una colección de personalizables[15] disponibles, una cantidad de moneda virtual[14] y su historial de partidas[8].

Partida[8] Iniciada: Estado de una partida[8] en curso. El estado incluye todos aquellos datos necesarios para la correcta reanudación de la partida[8]: Apuestas actuales, jugadores[12] participantes, turno actual y estado de las manos[9] y de la mesa.

Partida[8]: Código de invitación de la partida[8] y jugadores[12] que juegan en ella.

Personalizable[15]: Elemento estético desbloqueable que incluye todos los usuarios que lo poseen y tiene un precio.

Historial de Partidas[8]: El conjunto de partidas[8] jugadas por un jugador[12] donde se registran los jugadores[12] participantes, el ganador (o ganadores), la cantidad ganada por el ganador y cuando se inició la partida.

Glosario de Términos

Blackjack
Juego de cartas que sigue las normas del Blackjack Europeo.

Póker
Juego de cartas que sigue las normas del Póker Texas hold'em.

Sala
Espacio virtual donde se juntan jugadores[12].

Sala creada
Grupo de jugadores[12] de una sala[3] que aún no han iniciado ninguna partida[8] en esa sala[3].

Sala iniciada

Sala[3] donde el grupo de jugadores[12] ya está jugando una partida[8].

Sala privada
Sala[3] donde solo participan jugadores[12] invitados y su creador.

Sala pública
Sala[3] donde participan jugadores[12] invitados, su creador y jugadores[12] aleatorios hasta completar el aforo de la sala[3]. Este aforo varía según el juego[13].

Partida
Ciclo de juego[13] hasta que hay un ganador o ganadores. Este hecho inicia una nueva partida[8]. El fin de una partida[8] no inicia una nueva si el número de jugadores[12] de la sala[3] no permite jugar al juego[13].

Mano
Conjunto de cartas que un jugador[12] tiene en su posesión durante una partida[8].

Partida síncrona
Partida[8] que requiere a todos los jugadores[12] conectados al mismo tiempo.

Usuario autenticado
Usuario registrado que ha completado el proceso de autenticación con éxito.

Jugador
Usuario autenticado[11] que se encuentra en una sala[3].

Juegos
Blackjack[1] y Póker[2].

Moneda virtual
Moneda de cambio dentro de la aplicación con la que se realiza cualquier operación monetaria interna.

Personalizables
Elementos personalizables de un jugador[12]: su avatar, su apariencia de las cartas y su nombre de usuario.

ANEXO IV: FORMALIZACIÓN DE REQUISITOS

Requisitos Funcionales de los Usuarios No Autenticados

RFUNA-1. El sistema permitirá <Autenticarse>

i. con su <nombre de usuario> y <contraseña>.

RFUNA-2. El sistema permitirá <Registrarse>

i. con <una> <Cuenta de Google no registrada en el sistema>.

RFUNA-3. El sistema permitirá <Salir de la Aplicación>.

Requisitos Funcionales de los Usuarios Autenticados

RFUA-1. El sistema permitirá <Cerrar Sesión>.

RFUA-2. El sistema permitirá <Salir de la Aplicación>.

RFUA-3. El sistema permitirá <Crear> <Una>

a. <Sala Privada[6]> de <Blackjack[1]>.
b. <Sala Privada[6]> de <Póker[2]>.
c. <Sala Pública[7]> de <Blackjack[1]>.
d. <Sala Pública[7]> de <Póker[2]>.

RFUA-4. El sistema permitirá <Unirse> a <una> <Sala[3]>

a. <Creada[4]> como <Jugador[12]>

i. Con <un> <Código de Invitación>.

RFUA-5. El sistema permitirá

a. <Solicitar> la <Reanudación> de una <Sala Iniciada[5]> y
b. <Abandonar> una <Sala Iniciada[5]>

i. <Previamente> <Pausada>.

RFUA-6. El sistema permitirá <Cambiar>

a. <Nombre de Usuario>

i. Si <no existe otro> <Usuario> con el <nuevo> <Nombre de Usuario>.

b. <Avatar> y
c. <Apariencia de Cartas>

i. Si el <nuevo> está <Desbloqueado>.

RFUA-7. El sistema permitirá <Consultar> el <Historial de Partidas[8]>.

Requisitos Funcionales de los Jugadores

RFJ-1. El sistema permitirá

a. <Iniciar> <una> <Sala Pública[7] Creada[4]> de <Blackjack[1]>

i. De <1 hasta 7> <Jugadores[12]>.

b. <Iniciar> <una> <Sala Privada[6] Creada[4]> de <Blackjack[1]>

i. De <1 hasta 7> <Jugadores[12]>.

c. <Iniciar> <una> <Sala Pública[7] Creada[4]> de <Póker[2]>

i. De <1 a 10> <Jugadores[12]>.

d. <Iniciar> <una> <Sala Privada[6] Creada[4]> de <Póker[2]>

i. De <2 a 10> <Jugadores[12]>.

RFJ-2. El sistema permitirá <Abandonar> la <Sala Creada[4]>.

Requisitos Funcionales de los Jugadores de Blackjack

RFJB-1. El sistema permitirá

a. <Pausar> y

b. <Abandonar>

i. La <Sala[3]>.

RFJB-2. El sistema permitirá

a. <Pedir Carta> y

b. <Plantarse>

i. <Durante> su <Turno>,

ii. Si la <Mano[9]> no <supera 21> <Puntos>.

RFJB-3. El sistema permitirá <Apostar>

i. Al <Inicio> de una <Partida[8]>,

ii. Una <Cantidad> <entre 1 y el Saldo>.

Requisitos Funcionales de los Jugadores de Póker

RFJP-1. El sistema permitirá

a. <Pausar> y

b. <Abandonar>

i. La <Sala Iniciada[5]>.

RFJP-2. El sistema permitirá

a. <Igualar una Apuesta> y

b. <Subir una Apuesta>

i. En su <Turno>,

ii. A una <Cantidad> <entre 1 y el Saldo>.

iii. Cumpliendo las <Reglas de Apuestas del Póker[2]>.

c. <Retirarse>

i. En su <Turno>.

Requisitos No Funcionales

RNF-1. El sistema permitirá <Partidas síncronas[10]>.

RNF-2. El sistema permitirá <Autenticarse> en <distintos> <Dispositivos>

i. De forma <no simultánea>.

RNF-3. El sistema recompensará

a. A los <Ganadores> de una <Partida[8]> el <monto> de <Moneda Virtual[14]>

i. Respetando las reglas del juego[13].

b. A los <Jugadores[12]> con <500> <Monedas Virtuales[14]> al <Autenticarse>

i. No más de <una vez> en las últimas <24> <horas>.

RNF-4. El sistema deberá

a. <Mantener> la <Partida[8]> si <uno o más> <Jugadores[12]> <Pierden la Conexión> <hasta que> llegue el <Turno> de <alguno> de esos <Jugadores[12]>.

b. <Pausar> la <Partida[8]> si <un> <Jugador[12]> llega a su <Turno> <Sin Conexión>.

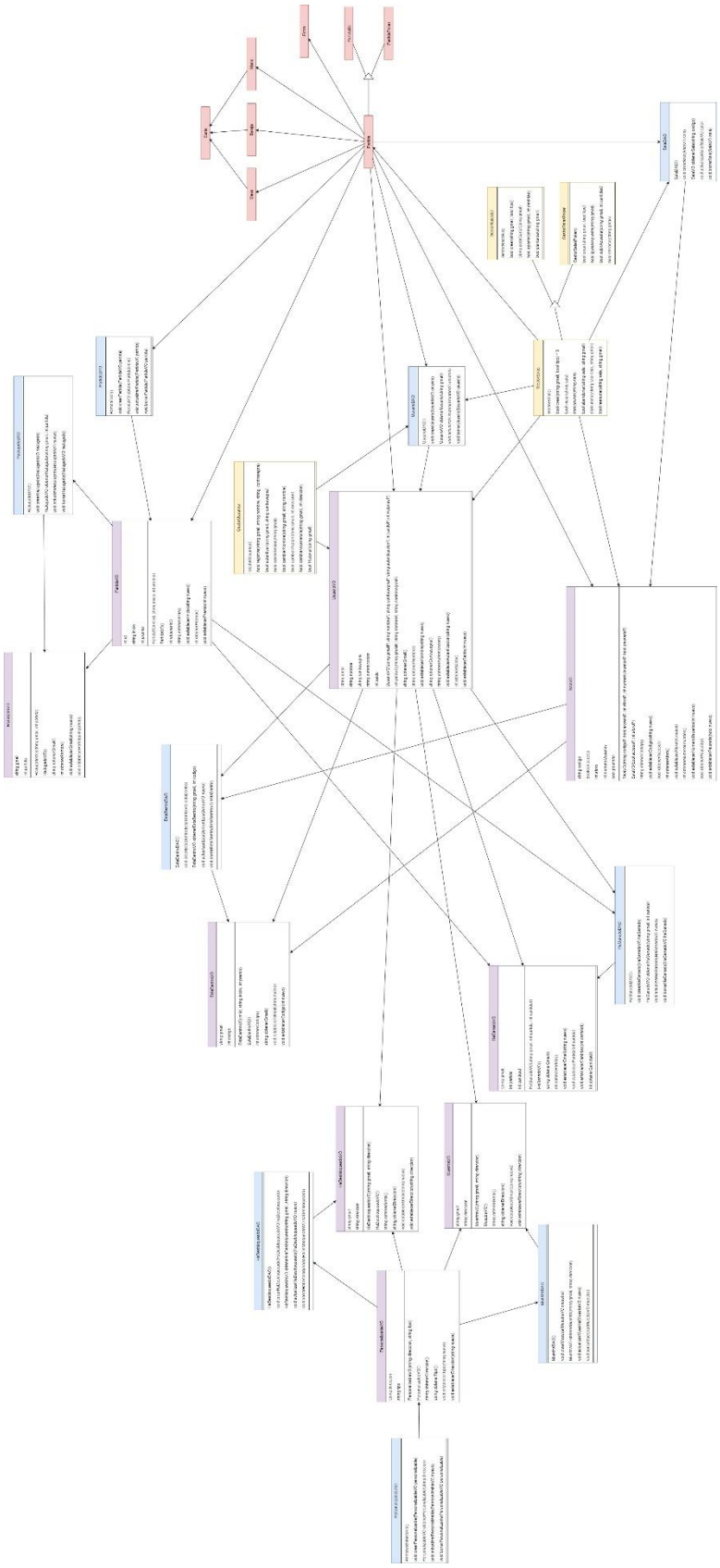
Restricciones

1. El sistema deberá <Desplegarse> en un <Dispositivo> con <Conexión a internet>

a. A través de un <Navegador Web>.

b. A través de una <Aplicación de Escritorio>.

ANEXO V: VISTA DE MÓDULOS DEL BACKEND



REFERENCIAS

[a] Web Content Accessibility Guidelines (WCAG) 2.0. W3C. Disponible en: [enlace]
[b] Golden Rules of User Interface Design - Ben Shneiderman. Disponible en: [enlace]
[c] Gestalt principles. Disponible en [enlace]