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: v0.0.1

CONTENIDO

1 INTRODUCCIÓN

<<....>>

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, ya 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, a parte 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

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.

Miembro del equipo	**Rol asignado**
López Ansón, Jesús	Principal contacto con los clientes. Responsable de GitHub. Encargado de dar la aprobación final a las tareas y documentos que se consideren terminados.
Bernad Ferrando, Lizer	Responsable de GitHub. Construcción del software.

	Responsable de tomar notas en las reuniones de los miembros del equipo o con los clientes.
Morales Rosa, Lucía	Responsable de apuntar las horas semanales trabajadas por cada miembro del equipo.
	Responsable de la asignación de tareas del backend.
	Responsable de la asignación de tareas del frontend para web.
	Responsable de la asignación de tareas del frontend para escritorio.
	Aseguramiento del producto (pruebas)
	Calendario del proyecto
Más??	

Tabla 2: Roles y responsabilidades

3 PLAN DE GESTIÓN DEL PROYECTO

En esta sección se va a detallar como 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. Se ha decidido emplear esta plataforma ya que ofrece hasta veinte gigabytes gratuitos de almacenamiento, más que suficiente para la base de datos que se va a implementar.

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 eran escasas.

Entre las opciones valoradas para el desarrollo 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 decidido 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.

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 habían empleado también Godot y consideraron que, en su experiencia, era más efectivo emplear esta tecnología.

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

En cuanto a la formación de los miembros del equipo se debe tener en cuenta que la mayoría de los integrantes no han empleado ninguna de las tecnologías mencionadas con anterioridad. Por ello, se ha decidido emplear las primeras semanas en 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.

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 NOMBRE.

En cuanto a la forma de nombrar los ficheros se ha decidido seguir convenciones generales de nombrado como el hecho de no poner espacios o acentos y el nombre comenzará con una letra minúscula. Para separar las distintas palabras se pondrá en mayúscula la primera letra de cada palabra y 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. Además, los nombres deberán ser cortos y describir el contenido del archivo. En cuanto al formato del código se ha decidido emplear la extensión "Prettier" de Visual Studio Code ya que todos empleamos este 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 responsables de realizar determinadas tareas como los merges 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 harán uso de las *labels* o etiquetas para añadir información a cada tarea. 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.

Métodos, herramientas y técnicas necesarios tanto para construir el software (p. ej. herramientas de desarrollo) como para desplegarlo y probarlo.

En cuanto a como realizar cambios en la documentación y código fuente, se creará una rama para cada nueva funcionalidad a implementar o problema a resolver, dejando en la rama principal el código lo más correcto y funcional posible. Para realizar los *merge* entre ramas, se deberá indicar con una incidencia a los responsables de GitHub que la rama se ha finalizado y esta 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 titulo descriptivo y una descripción que detalle los cambios realizados.

• Cómo se construye e integra el software: si hay scripts de construcción automatizada o no (en ese caso qué se usa, y cómo se garantiza que todos los participantes compilan igual y con las mismas dependencias), qué se incluye en la construcción (descarga y actualización de dependencias, compilación, ejecución de tests automáticos...) y cada cuánto se construye (compila, integra, prueba) el sistema completo, cómo se configuran los computadores de los desarrolladores.

• Cómo se despliega el software más allá de las máquinas de desarrollo: contenedores, máquinas virtuales, servidor en cloud etc. y cómo se configuran esos entornos (rutas, usuarios y contraseñas, puertos y otros elementos).

• Guías para la documentación de diseño del software y otros documentos del proyecto, guías para el diseño gráfico de las GUI (estética, usabilidad...).

• Actividades de control de calidad del código que se realizarán: revisiones de código por pares, revisiones de requisitos o diagramas UML por pares, tipos de tests automáticos o manuales que se llevarán a cabo.

• Cómo se hará la entrega de resultados a clientes.

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

COMPLETAR LOS TRES PRIMEROS PUNTOS.

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 modulo 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 como 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].
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.
c.	El sistema recompensará a los jugadores[12] con un personalizable[15] tras ganar 10 partidas[8] si existe algún personalizable[15] no desbloqueado.
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 deberá pausar la partida[8] si un jugador[12] llega a su turno sin conexión.

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

Diagrama de módulos, componente y conector y de distribución.

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.

El lenguaje de programación elegido es 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, como los integrantes del grupo de backend posteriormente ayudarán en frontend, se ha decidido que también comiencen a usar TypeScript para familiarizarse con él.

Para el desarrollo del frontend de escritorio se ha optado por emplear ESCRITORIO ya que…

si hay una API Web va a ser REST[ful] o no, si algunas de las operaciones van a ser asíncronas o no, si va a ser una aplicación móvil o de escritorio será nativa o se van a usar

tecnologías web, cómo se van a considerar los requisitos de seguridad o de prestaciones, cómo y dónde se harán las instalaciones y despliegues etc.)

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, imprevistos surgidos, etc.>>

5.1 Inicio del proyecto

<<...>>

5.2 Ejecución y control del proyecto

<<...>>

5.3 Cierre del proyecto

<<...>>

6 CONCLUSIONES

<<...>>

GLOSARIO

ANEXO I. PRESUPUESTO

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 la misma.

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.

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>.

c. A los <Jugadores[12]> con <un> <Personalizable[15]>

i. <Tras> <Ganar> <10> <Partidas[8]>,

ii. Si <Existe> <algún> <Personalizable[15]> <no desbloqueado>.

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>.

WEBGRAFÍA