Universidad de Valladolid

ESCUELA DE INGENIERÍA INFORMÁTICA (SG)

Grado en Ingeniería Informática de Servicios yAplicaciones

Juegos de cartas online: Algoritmos criptográficos

Alumno(a): Joseba Ramos Martínez

Tutor(a): José Ignacio Farrán Martín

Trabajo Fin de Grado

Grado en Ingeniería informática de servicios y aplicaciones

Juegos de cartas online: algoritmos criptográficos

Autor: Joseba Ramos Martínez

Tutor: José Ignacio Farrán Martín

Índice

1 Introducción

1.1 Motivación

Ante el aumento del número de jugadores de poker en estos últimos años, se desea crear una aplicación empresarial "JPOKER" para gestionar una aplicación web que permite, principalmente, a los usuarios (jugadores) jugar partidas, crear y unirse a partidas privadas y jugar contra el ordenador.

1.2 Alcance

Debido a que el número de usuarios va a ser elevado, a que se van a producir transacciones monetarias (en un futuro), y que van a contener información sensible de los usuarios, es fundamental asegurar la seguridad del sistema, la escalabilidad debido al aumento de usuarios y la fiabilidad.

A medio plazo también se espera que la aplicación pueda extenderse al ámbito internacional, lo cual implicaría una notable mejora en la calidad y fluidez de la aplicación.

1.3 Objetivos

Los objetivos principales de la aplicación desarrollada son las siguientes:

ID	**Objetivo**
OBJ-01	Jugar partidas de poker
OBJ-02	Crear partidas privadas de poker
OBJ-03	Unirse a partidas privadas de poker
OBJ-04	Jugar partidas de poker contra el ordenador
OBJ-05	Aplicación de algoritmos y protocolos criptográficos necesitados para implementar juegos de cartas online.
OBJ-06	Administración de jugadores.
OBJ-07	Administración de partidas.

1.4 Organización del documento

Este contenido de este documento ha sido organizado en los siguientes puntos principales:

- Estado del Arte
- Herramientas utilizadas
- Estimación
- Análisis
- Diseño
- Algoritmo de cifrado
- Pruebas
- Manuales
- Conclusiones

2 Estado del arte

En este apartado se estudian los diferentes programas software del mercado actual que proveen una funcionalidad similar a nuestro sistema.

Existen multitud de programas software para gestión de páginas de juegos de cartas y apuestas, los cuales proveen una funcionalidad similar y, obviamente, más amplia que la de nuestra aplicación empresarial.

2.1 Propuestas similares a la aplicación empresarial desarrollada

En este caso, vamos a centrarnos en 3 aplicaciones específicas: pokerstars, 888poker y bet365.

1. **Pokerstars**
   Es una sala de póquer en línea propiedad de The Stars Group. Se puede acceder a través de clientes de póker descargables para Windows, macOS, Android e iOS. Es el sitio de póker en línea con dinero real más grande del mundo y controla más de dos tercios del mercado total de póker en línea.
2. **888poker**
   Es una red de póquer en línea fundada en 1997. Forma parte de un grupo de marcas de entretenimiento en línea propiedad de 888 Holdings plc..
3. **bet365**
   Bet365 es una compañía dedicada al juego en línea que ofrece apuestas deportivas, póquer, juegos de casino y bingo, así como vídeo en directo relacionado con eventos deportivos.

- Ventajas

pokerstars	**888poker**	**bet365**
4 jugadores máximo	3 jugadores máximo	Distintos modos de juego
Gestiona el juego responsable	Precio fijo	Gestiona el juego responsable
Rápido reintegro	Gestiona el juego responsable	Alertas de actividad

- Desventajas

pokerstars	**888poker**	**bet365**
Software de pago	No rápido reintegro	
	Software de pago	

3 Herramientas utilizadas

Para la realización de este proyecto se han utilizado las siguientes herramientas:

3.1 Visual Studio Code

Es un editor de código fuente desarrollado por Microsoft, Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código. También es personalizable, por lo que los usuarios pueden cambiar el tema del editor, los atajos de teclado y las preferencias. Es gratuito y de código abierto, aunque la descarga oficial está bajo software privativo e incluye características personalizadas por Microsoft.

Visual Studio Code se basa en Electron, un *framework* que se utiliza para implementar Chromium y Node.js como aplicaciones para escritorio, que se ejecuta en el motor de diseño Blink. Aunque utiliza el *framework* Electron, el software no usa Atom y en su lugar emplea el mismo componente editor (Monaco) utilizado en Visual Studio Team Services (anteriormente llamado Visual Studio Online).

3.2 MYSQL

MySQL es un sistema de gestión de bases de datos relacional desarrollado bajo licencia dual: Licencia pública general/Licencia comercial por Oracle Corporation y está considerada como la base de datos de código abierto más popular del mundo, y una de las más populares en general junto a Oracle y Microsoft SQL Server, todo para entornos de desarrollo web.

Está desarrollado en su mayor parte en ANSI C y C++. Tradicionalmente se considera uno de los cuatro componentes de la pila de desarrollo LAMP y WAMP.

3.3 PYTHON

Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código. Se trata de un lenguaje de programación multiparadigma, ya que soporta parcialmente la orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, dinámico y multiplataforma.

Administrado por la Python Software Foundation, posee una licencia de código abierto, denominada Python Software Foundation License. Python se clasifica constantemente como uno de los lenguajes de programación más populares.

3.4 HTML

HTML, siglas en inglés de *HyperText Markup Language*, hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código ara la definición de contenido de una página web, como texto, imágenes, videos, juegos, entre otros.

3.5 CSS

CSS (siglas en inglés de Cascading Style Sheets) es un lenguaje de diseño gráfico para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado. Es muy usado para establecer el diseño visual de los documentos web, e interfaces de usuario escritas en HTML o XHTML; el lenguaje puede ser aplicado a cualquier documento XML, incluyendo XHTML, SVG, XUL, RSS, etcétera. Junto con HTML y JavaScript, CSS es una tecnología usada por muchos sitios web para crear páginas visualmente atractivas, interfaces de usuario para aplicaciones web y GUIs para muchas aplicaciones móviles.

CSS está diseñado principalmente para marcar la separación del contenido del documento y la forma de presentación de este, características tales como las capas o *layouts*, los colores y las fuentes. Esta separación busca mejorar la accesibilidad del documento, proveer más flexibilidad y control en la especificación de características presentacionales, permitir que varios documentos HTML compartan un mismo estilo usando una sola hoja de estilos separada en un archivo .css, y reducir la complejidad y la repetición de código en la estructura del documento.

3.6 JAVASCRIPT

JavaScript es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

Se utiliza principalmente del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas y JavaScript del lado del servidor. Su uso en aplicaciones externas a la web es también significativo.

3.7 GitHub

GitHub es una plataforma de desarrollo colaborativo para alojar proyectos utilizando el sistema de control de versiones Git. Se utiliza principalmente para la creación de código fuente de programas de ordenador. El software que opera GitHub fue escrito en Ruby on Rails. El código de los proyectos alojados en GitHub se almacena típicamente de forma pública.

4 Estimación

A continuación, se realizarán dos estimaciones del proyecto, una mediante puntos de caso de uso y otra mediante puntos de función.

4.1 Estimación mediante puntos de casos de uso

Este método está basado en las técnicas de puntos de función.

Un caso de uso es la descripción de una acción o actividad. Un diagrama de caso de uso es una descripción de las actividades que deberá realizar alguien o algo para llevar a cabo algún proceso

Los casos de uso no abordan la complejidad de las funciones ni de las características que se describen. El esfuerzo de desarrollo de un caso de uso depende mucho del desarrollador.

A partir de las especificaciones dadas por el cliente hemos sacado los siguientes casos de uso, representados en el diagrama expuesto a continuación:

Cálculo de la estimación por casos de uso

Primero hay que clasificar cada interacción entre cada actor y caso de uso según su complejidad y le asignamos un peso.

Tipo de interacción	**Peso**
Simple (a través de la API)	1
Medio (a través de un protocolo)	2
Complejo (a través de una interfaz gráfica)	3

Todas las interacciones del actor con el sistema se llevan a cabo a través de una interfaz gráfica por lo que el peso de todas ellas será de 3.

Tabla de actores

Actor	**Peso**
Usuario	3
Administrador	3

Procedemos a calcular la complejidad de cada caso de uso según el número de transacciones o escenarios.

Tipo de CU	**Número de transacciones**	**Peso**
Simple	3 o menos	5
Medio	De 4 a 6	10
Complejo	7 o más	15

Por lo tanto, el peso de los casos de uso será:

Caso de Uso	**Complejidad**	**Peso**
Login	Simple	5
Registrarse	Simple	5
Ver Ranking ganadores	Simple	5
Mostrar reglas	Simple	5
Jugar partida	Medio	10
Crear partida	Medio	10
Unirse partida privada	Medio	10
Partida contra el ordenador	Simple	5
Ver perfil de usuario	Simple	5
Apostar	Simple	5
Pasar	Simple	5
No ir	Simple	5
Igualar	Simple	5
Subir Apuesta	Simple	5
Pedir carta	Simple	5
Descartar carta	Simple	5
Ver Ganadores	Simple	5
Administrar jugadores	Simple	5
Administrar Partidas	Simple	5
Registrar administradores	Simple	5
Total	**115**	

Ahora, calculamos los puntos de caso de uso sin ajustar:

UUCP = peso de los actores sin ajustar + peso de los casos de uso sin ajustar

UUCP = UAW + UUCW = 10 + 115 = 125 Puntos de casos de uso sin ajustar

Calculamos el Factor de Complejidad Técnica (TCF)

- Resultado = Peso * Influencia
- Influencia:

0-2	Irrelevante
2-4	Media
5	Esencial

Factor	Descripción	Peso	Influencia	Resultado	Justificación
R1	Sistema distribuido	2	3	2*3 = 6	Es un sistema con versión web por lo que sí posee un cierto nivel de distribución
R2	Objetivos de rendimiento	1	1	1*1=1	
R3	Eficiencia respecto al usuario final	1	1	1*1=1	
R4	Procesamiento complejo	1	1	1*1=1	
R5	Código reutilizable	1	3	1*1=1	
R6	Instalación sencilla	0.5	5	0,5*5 =2.5	Sencillez para facilitar el despliegue
R7	Fácil utilización	0.5	4	0,5*4 = 2	Amigable y fácil de usar
R8	Portabilidad	0.5	4	0.5*4=2	Funcionará vía en plataformas web
R9	Fácil de cambiar	1	5	1*5=5	Esencial debido a la volatilidad
R10	Uso concurrente	1	3	1*3=3	
R11	Características de seguridad	2	3	2*3=6	Cifrado de las cartas durante la partida
R12	Accesible por terceros	0.5	5	0.5*5=2.5	Los usuarios acceden a través de una plataforma web
R13	Se requiere formación especial	0.5	0	0,5*0 = 0	No se requiere formación alguna, fácil de usar

$$TCF = 0{,}6 + (0{,}01 * \sum_{I=1}^{13} Ri)$$

Por lo tanto,

TCF = 0,6 + (0.01*(6+1+1+1+1+2.5+2+2+2.5+3+6+5+0)) = 0,6 + (0.01*33) = 0.93

Cálculo del Factor Entorno (EF)

Factor	Descripción	Peso	Influencia	Resultado	Justificación
R1	Familiar con el modelo proyecto	2	4	2*4 = 8	Se ha trabajado en más proyectos software
R2	Experiencia en la aplicación	0,5	4	0,5*4 = 2	Con cierta experiencia en aplicaciones similares
R3	Experiencia en orientación a objetos	1	5	1*4=4	Alta experiencia de todos los miembros del equipo
R4	Capacidades de análisis	0,5	4	0,5*4 = 2	El equipo ha realizado análisis en otros proyectos software
R5	Motivación	1	5	1*5=5	Motivación máxima
R6	Requisitos estables	2	1	2*1 = 2	Se prevé una alta volatilidad y cambios en los requisitos
R7	Trabajadores a tiempo parcial	-1	5	1*-5 = -5	Compatibilizar con otros trabajos
R8	Lenguaje complejo	-1	1	-1*1 = -1	Se usará el lenguaje Python

$$EF = 1.4 + (-0.03 * \sum_{i=1}^{8} Ri\ )$$

Por lo tanto,

EF = 1.4 + (-0.03*(8+2+4+2+5+2-5-1)) = 1.4 + (-0.03*17) = 1.4 – 0.51 = 0.89

Calculamos a continuación los puntos de caso de uso ajustados

(UCP): UCP = UUCP * TCF * EF = 125 * 0.93 * 0.89 = 103.5 **Puntos de casos de uso ajustados**

Procedemos a calcular estimación del esfuerzo:

Esfuerzo = UCP * Factor de Productividad = 103.5 * 20 horas/persona = 2069 horas/persona

El factor de productividad utilizado es el de Karner, que establece que se necesitan 20 horas por persona en por cada punto de función.

Estimación esfuerzo proyecto completo:

Actividad	**% proyecto**	**Horas x persona**
Análisis	10	517.25
Diseño	20	1034.5
Programación	40	2069
Pruebas	15	775.875
Sobrecarga	15	775.875
Total	5172	

El esfuerzo estimado en distintas unidades sería:

Esfuerzo (en meses) = 5172 horas-persona / 8 horas-día / 21 días-mes = 30.78 meses

Teniendo en cuenta que una jornada laboral dura 8 horas al día y un mes cuenta con 21 días laborales de media. Teniendo en cuenta que el proceso puede ser paralelizado, gracias a disponer de 4 miembros en el equipo de trabajo, el tiempo total sería:

Tiempo (en meses) = 30.78 meses / 4 personas = 7.69 meses

4.2 Estimación por puntos de función

La estimación mediante puntos de función es una técnica basada en una métrica que cuantifica la funcionalidad (en relación con su complejidad) que se debe entregar al usuario al construir la aplicación.

Para la realización de los cálculos de la estimación por puntos de función se han seguido la siguiente tabla de complejidades:

Parámetro	**Complejidad Baja**	**Complejidad Media**	**Complejidad Alta**
Entradas	3	4	6
Salidas	4	5	7
Consultas	3	4	6
Ficheros internos	7	10	15
Ficheros externos	5	7	10

Para comenzar el proceso de estimación, calculamos el número de entradas, salidas, consultas, ficheros internos y externos, junto a sus elementos.

ENTRADAS		
Nombre	**Complejidad**	
Introducir datos de registro en el sistema	Baja	3
Introducir login en el sistema	Baja	3
Seleccionar número de jugadores de la partida privada	Baja	3
Introducir código partida	Baja	3
Introducir apuesta inicial	Baja	3
Seleccionar opción ronda	Media	4
Realizar apuesta	Baja	3
Seleccionar pedir o no carta	Baja	3
Seleccionar carta descartada	Baja	3

SALIDAS		
Nombre	**Complejidad**	
Mostrar opciones en rondas	Baja	4
Mostrar cartas	Baja	4
Mostrar ganadores	Baja	4
Mostrar perfil	Baja	4
Mostrar tabla top ganadores	Baja	4
Mostrar Reglas	Baja	4

CONSULTAS		
Nombre	**Complejidad**	
Listar top 10 ganadores con más victorias	Baja	3
Listar datos usuario registrado	Baja	3
Identificar usuario	Baja	3
Identificar partida	Baja	3
Identificar partidas disponibles	Baja	3
Comprobar datos partida	Baja	3
Actualizar datos partida	Baja	3

FICHEROS LÓGICOS INTERNOS		
Nombre	**Complejidad**	
Tabla de usuarios	Media	10
Tabla de partidas	Baja	7

FICHEROS EXTERNOS		
Nombre	**Complejidad**	
Django users	Media	7

Procedemos a calcular de puntos de función no ajustados:

Componentes	**Total**	
Entradas	8 (Baja) + 1 (Media)	28
Salidas	6 (Baja)	24
Consultas	7 (Baja)	21
Ficheros internos	1(Baja) + 1 (Media)	17
Ficheros externos	1 (Media)	7
Total	**97**	

A continuación, en la siguiente tabla se muestra los factores de ajuste que se usan para calcular los **Puntos de Función Ajustados (PFA)**

En esta tabla, a cada atributo se le asignará un valor entre 0 y 5, según su grado de influencia:

Grado de influencia	**Valor**
Sin influencia	0
Influencia mínima	1
Influencia moderada	2
Influencia apreciable	3
Influencia significativa	4
Alta Influencia	5

Tabla de factores

Factor	**Valor**
1. Comunicación de Datos	4
2. Funciones Distribuidas	2
3. Prestaciones	1
4. Gran uso de la configuración	2
5. Velocidad de las transacciones	3
6. Entrada de datos en línea	4
7. Diseño para la eficiencia del usuario final	4
8. Actualización de datos en línea	3
9. Complejidad del proceso lógico interno de la aplicación	2
10. Reusabilidad del código por otras aplicaciones	0
11. Facilidad de instalación	1
12. Facilidad de operación	0
13. Localizaciones múltiples	2
14. Facilidad de cambios	2
Suma	31

$FA = 0{,}65 + 0{,}01 \cdot \sum \text{Factores Complejidad} = 0.65 + 0.01 \cdot 31 = 0.96$

Con estos valores obtenidos podemos obtener los puntos de función ajustados (PFA):

PFA = PFNA · FA = 97 · 0.96 = 93.12

Considerando que vamos a utilizar el lenguaje Python para el desarrollo de la aplicación, consultamos el número de líneas de código promedio por Punto de Función que establece la industria del software cuando se utiliza Python.

En Python, un Punto de Función (PF) equivale a 62 líneas de código (LDC).

Por lo tanto, el tamaño del software en líneas de código será: Tamaño Software = 93.12 PF · 62 LDC/PF = 5773.44 LDC = 5.77 KLDC

Utilizando el método de COCOMO II podemos estimar el esfuerzo necesario y el coste total del proyecto:

Al realizar la estimación al comienzo del proyecto, se utilizará el modelo COCOMO Básico con modo orgánico, al desarrollarse la aplicación software en un entorno estable, con poca innovación técnica y haberse estimado menos de 50 KLDC.

Esfuerzo = $2.4 \cdot 5.77^{1.05}$= 15.11 personas-mes

Sabiendo que somos 4 personas en el equipo de trabajo:

Tiempo Desarrollo = 15.11 / 4 = 4.78 meses

Para hallar el coste total del proyecto hallamos la media de los sueldos de los integrantes del equipo de trabajo:

Miembro	**Sueldo mensual**
Miembro 1	2200 €
Miembro 2	1530 €
Miembro 3	1220 €
Miembro 4	1220 €
Media	**1542.5 €**

Por lo tanto, el coste estimado del proyecto será:

Coste Desarrollo = (2200€/mes + 1530€/mes + 1220€/mes +1220€/mes) * 4.78 meses = 29492.6 €

4.3 Comparativa

Tras realizar las estimaciones mediante dos procedimientos distintos, casos de uso y puntos de función, podemos observar que los resultados no coinciden:

Estimación	**Casos de uso**	**Puntos de función**
Personas/mes	30.78	15.11
Meses con 4 desarrolladores	7.69	4.78

Para reconciliar la estimación por puntos de caso de uso, realizada anteriormente, con esta estimación por puntos de función calculamos la media entre ambas:

Duración Proyecto = (7.69 + 4.78) / 2 = 6.2 meses

Por lo tanto, el coste estimado del proyecto será:

Coste Desarrollo = (2200€/mes + 1530€/mes + 1220€/mes + 1220€/mes) * 6.2 meses = 38254€

4.4 Presupuesto

A menudo es conveniente desglosar los costes estimados a lo largo del proyecto, para ofrecer una información más detallada de la distribución de los recursos de cara a la dirección.

El coste relacionado con la adquisición de Hardware y Software preciso para el desarrollo, implantación y funcionamiento normal del sistema es el siguiente:

	Precio	**Tiempo vida útil**	**Tiempo proyecto**	**Prorrateo/coste**	**Uso en el proyecto**
Portátil	500€ (por 4 unidades)	5 años	5.9 meses	500*0.39/5 = 39€	12,82%
Impresora	100€	3 años	5.9meses	100*0.39/3 = 13€	7.69%
Conexión Internet	50€/mes		5.9 meses	50 * 4.69 = 234.5€	
Firebase	500€/mes	1 año	5.9 meses	500*12= 5000€	
Instalación	100€		1 semana	100€	
Mantenimiento	250€		1 año	400*12= 4800€	
Gastos oficina	1000€		5.9 meses	250 €	

Por lo tanto, el presupuesto total del proyecto será:

	Coste
Hardware	1600€
Software	9436.5€
Desarrollo	36403€
Recursos	1350€
Total	**44640.5€**

Nota: debemos tener en cuenta que el presupuesto de un proyecto no deja de ser una estimación y puede no coincidir con el coste total del proyecto.

5 Análisis

5.1 Descripción de actores

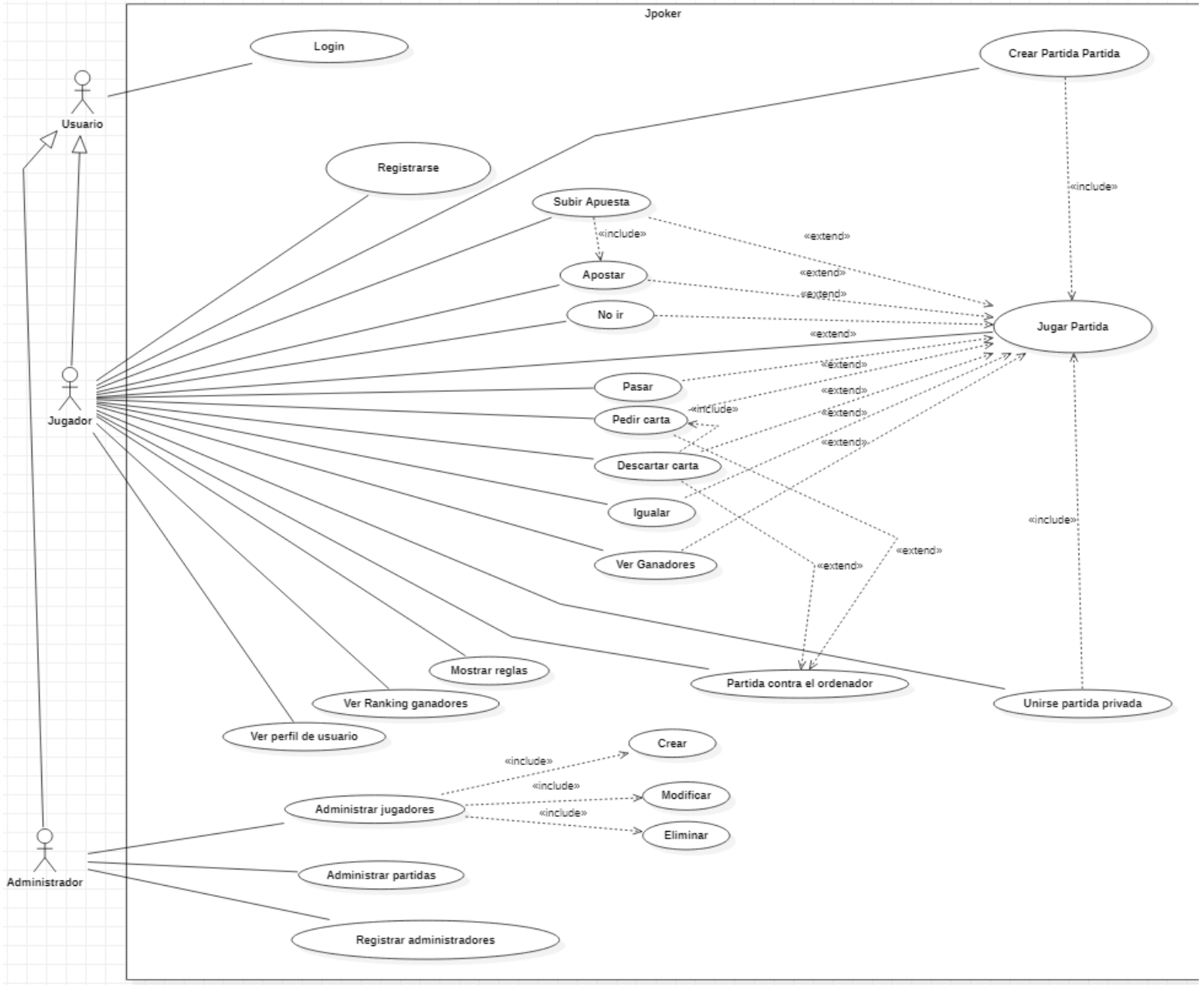
ID	Actor	Descripción
AC-01	Jugador	Este actor representa a los usuarios que juegan las partidas
Ac-02	Administrador	Este actor representa a los usuarios que administran la página

5.2 Requisitos de usuario

Se han abstraído los siguientes casos de uso:

ID	Caso de Uso
CU-01	Login
CU-02	Registrarse
CU-03	Ver Ranking ganadores
CU-04	Mostrar reglas
CU-05	Jugar partida
CU-06	Crear partida
CU-07	Unirse partida privada
CU-08	Partida contra el ordenador
CU-09	Ver perfil de usuario
CU-10	Apostar
CU-11	Pasar
CU-12	No ir
CU-13	Igualar
CU-14	Subir Apuesta
CU-15	Pedir carta
CU-16	Descartar carta
CU-17	Ver Ganadores
CU-18	Administrar jugadores
CU-19	Administrador de partidas
CU-20	Registrar administrador

5.3 Diagrama Casos de Uso


Jpoker
Login
Crear Partida Partida
Usuario
Registrarse
Subir Apuesta
«include»
«extend»
Apostar
«extend»
«extend»
No ir
Jugar Partida
«extend»
Jugador
Pasar
«extend»
«extend»
Pedir carta
«include»
«extend»
«extend»
Descartar carta
Igualar
«include»
Ver Ganadores
«extend»
«extend»
Mostrar reglas
Partida contra el ordenador
Unirse partida privada
Ver Ranking ganadores
Ver perfil de usuario
Crear
«include»
«include»
Administrar jugadores
Modificar
«include»
Eliminar
Administrador
Administrar partidas
Registrar administradores

5.4 Especificación de los Casos de Uso

CU-02	Partida contra el ordenador
Descripción	Registro de un nuevo usuario en la aplicación.
Precondición	Usuario no registrado
Secuencia Normal	1- Acceder al botón “Registrarse” 2- Añadir los datos 3- Darle a enviar para añadir un nuevo usuario
Postcondición	Usuario registrado
Excepciones	2.1- A la hora de añadir algún dato, este sea erróneo Sol: El sistema no permite añadir el dato y he indique el error 3.1- Usuario existente Sol: El sistema no permitirá añadir al usuario y mostrará un mensaje
Rendimiento	Medio
Frecuencia	Media
Importancia	Media
Comentarios	

CU-05	Jugar partida
Descripción	El usuario se dispondrá a jugar una partida de poker contra otros jugadores
Precondición	Usuario registrado en la aplicación
Secuencia Normal	1- Acceder al botón “jugar partida” 2- Entra en partida 3- Realiza apuesta inicial 4- Juega la ronda 2 4.1- Apuesta 4.2- Pasar 4.3- No ir 4.4- Iguala 4.5- Subir apuesta 5- Juega ronda 3 5.1- Pedir carta 5.1.2- Descartar carta 6- Juega ronda 4 6.1- Apuesta 6.2- Pasar 6.3- No ir 6.4- Igualar 6.5- Subir apuesta 7- Ver los ganadores
Postcondición	Usuario juega y finaliza la partida

Excepciones	1.1- No tener monedas 10 monedas mínimas para apostar Sol: Redirigir a página de inicio y mensaje de error 4.1.1- No tener monedas para apostar Sol: Mensaje de error y otra dar posibilidad de otra opción de la ronda 2 4.1.4- No poder igualar la apuesta más alta Sol: Mensaje de error y otra dar posibilidad de otra opción de la ronda 2 4.1.5- No poder igualar o subir la apuesta más alta Sol: Mensaje de error y otra dar posibilidad de otra opción de la ronda 2 6.1.1- No tener monedas para apostar Sol: Mensaje de error y otra dar posibilidad de otra opción de la ronda 4 6.1.4- No poder igualar la apuesta más alta Sol: Mensaje de error y otra dar posibilidad de otra opción de la ronda 4 6.1.5- No poder igualar o subir la apuesta más alta Sol: Mensaje de error y otra dar posibilidad de otra opción de la ronda 4
Rendimiento	Alto
Frecuencia	Alta
Importancia	Alta
Comentarios	

CU-06	Crear partida
Descripción	El usuario creará una partida privada y compartirá el código de la partida
Precondición	Usuario registrado en la aplicación
Secuencia Normal	1- Acceder al botón “Crear partida privada” 2- Añadir número de jugadores 3- Acceder al botón “Entrada partida” 4- Jugar Partida (CU-05)
Postcondición	Usuario juega y finaliza la partida
Excepciones	1.1- No tener monedas 10 monedas mínimas para apostar Sol: Redirigir a página de inicio y mensaje de error 2.1- No añadir número jugadores Sol: Mensaje añadir número de jugadores
Rendimiento	Alto
Frecuencia	Alta
Importancia	Alta

CU-07	Unirse partida privada
Descripción	El unirá creará una partida privada añadiendo el código de la partida
Precondición	Usuario registrado en la aplicación
Secuencia Normal	1- Acceder al botón "Jugar partida privada" 2- Introducir el código de la partida 3- Acceder al botón "Enviar" 4- Jugar Partida (CU-05)
Postcondición	Usuario juega y finaliza la partida
Excepciones	1.1- No tener monedas 10 monedas mínimas para apostar Sol: Redirigir a página de inicio y mensaje de error 2.1- No existir partida con dicho código Sol: Mensaje no existencia partida con dicho código
Rendimiento	Alto
Frecuencia	Alta
Importancia	Alta
CU-08	Partida contra el ordenador
Descripción	El usuario se dispondrá a jugar una partida de poker contra el ordenador
Precondición	Usuario no registrado
Secuencia Normal	1- Acceder al botón "Partida contra el ordenador" 2- Entra en partida 3- Pedir carta 3.1- Descartar carta 4- Ver los ganadores
Postcondición	Usuario registrado
Excepciones	1.1- No tener monedas 50 monedas mínimas para apostar Sol: Redirigir a página de inicio y mensaje de error
Rendimiento	Medio
Frecuencia	Media
Importancia	Media
Comentarios	

5.5 Requisitos funcionales y no funcionales

Un requisito funcional define una función del sistema de software o sus componentes, es decir, describen un conjunto de entradas, comportamientos y salidas.

ID	**CU-03 Ver Ranking ganadores**
RF-01	El sistema permitirá ver el top 10 de jugadores con más victorias

ID	**CU-05 Jugar Partida**
RF-01	El sistema permitirá realizar una apuesta inicial
RF-02	El sistema permitirá ver mis cartas
RF-03	El sistema permitirá ver las opciones que hay durante las rondas
RF-04	El sistema permitirá a los usuarios, en caso de no poder hacer una de las opciones, realizar otra
RF-05	El sistema permitirá pedir carta
RF-06	El sistema permitirá descartar carta
RF-07	El sistema permitirá ver los ganadores

ID	**CU-06 Crear Partida**
RF-01	El sistema permitirá crear una partida privada
RF-02	El sistema redirigirá mostrará el código de la partida para compartirla con el resto de los jugadores
RF-03	El sistema redirigirá al jugador a la partida

ID	**CU-07 Unir Partida**
RF-01	El sistema permitirá unirse a una partida privada
RF-02	El sistema permitirá introducir el código de la partida
RF-03	El sistema redirigirá al jugador a la partida

ID	**CU-08 Partida contra el ordenador**
RF-01	El sistema permitirá unirse a una partida contra el ordenador
RF-02	El sistema permitirá pedir una carta para mejorar la mano
RF-03	El sistema permitirá descartar una carta
RF-04	El sistema permitirá ver los ganadores

ID	**CU-09 Ver perfil de usuario**
RF-01	El sistema permitirá ver el perfil de usuario del jugador donde se mostrará en Login, las monedas y las victorias

Un requisito no funcional o atributo de calidad es un requisito que especifica criterios que pueden usarse para juzgar la operación de un sistema, es decir, describen características de funcionamiento.

ID	**RNF-01**
Definición	Tiempo de Respuesta Aceptable
Autor	Joseba
Descripción	El sistema mostrará la información al usuario en un tiempo de respuesta inferior a 5 segundos en el 90% de las ocasiones.
Importancia	Muy alta
Comentarios	En partida el tiempo de respuesta será casi instantáneo

ID	**RNF-02**
Definición	Consistencia e Integridad de los datos
Autor	Joseba
Descripción	El almacenamiento de la información será consistente y garantizará la integridad de los datos.
Importancia	Alta
Comentarios	

ID	**RNF-03**
Definición	Robustez
Autor	Joseba
Descripción	El sistema presentará robustez frente a fallos.
Importancia	Muy alta
Comentarios	

ID	**RNF-04**
Definición	Escalabilidad
Autor	Joseba
Descripción	El sistema será fácilmente escalable ante un incremento en la demanda a medio plazo
Importancia	Alta
Comentarios	

ID	**RNF-05**

Definición	Reutilización de software
Autor	Joseba
Descripción	El sistema fomentará el uso de componentes de software reutilizables.
Importancia	Alta
Comentarios	

ID	**RNF-06**
Definición	Usabilidad
Autor	Joseba
Descripción	La interfaz de usuario facilitará la usabilidad de la aplicación para los usuarios
Importancia	Alta
Comentarios	

ID	**RNF-07**
Definición	Alta fiabilidad
Autor	Joseba
Descripción	El sistema será fiable frente a caídas del sistema.
Importancia	Alta
Comentarios	

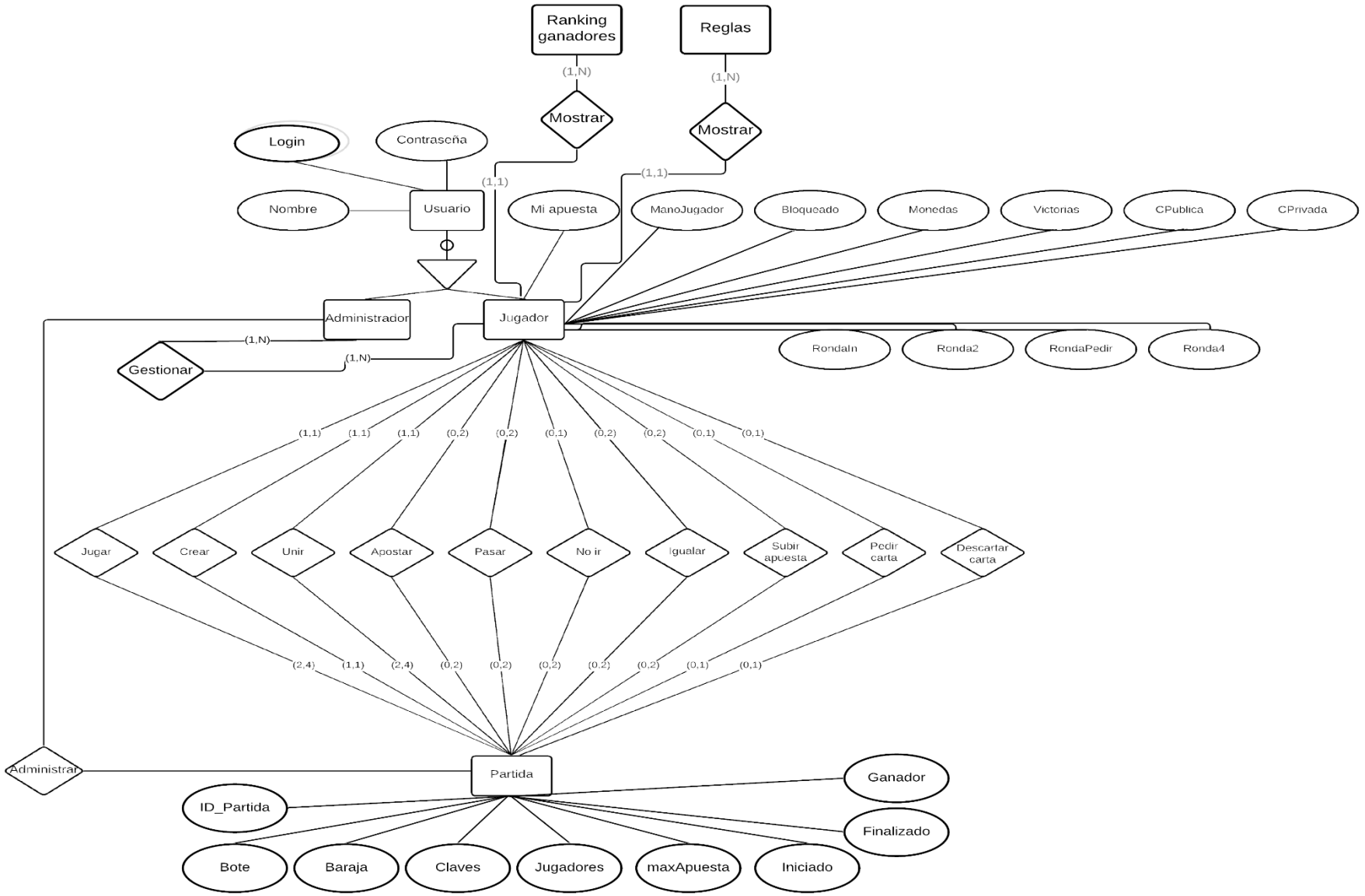
ID	**RNF-08**
Definición	Alta fiabilidad
Autor	Joseba
Descripción	El sistema será fiable frente a caídas del sistema.
Importancia	Alta
Comentarios	

5.6 Requisitos de información

Los requisitos de información describen todos los aspectos relacionados con los datos con los que opera el sistema.

- Modelo Conceptual (Entidad - Relación)

El modelo Entidad Relación nos facilita una representación de la base de datos dará soporte de almacenamiento a nuestra aplicación


Ranking ganadores
Reglas
(1,N)
(1,N)
Mostrar
Mostrar
Login
Contraseña
(1,1)
(1,1)
Nombre
Usuario
Mi apuesta
ManoJugador
Bloqueado
Monedas
Victorias
CPublica
CPrivada
Administrador
Jugador
(1,N)
(1,N)
Gestionar
RondaIn
Ronda2
RondaPedir
Ronda4
(1,1) (1,1) (1,1) (0,2) (0,2) (0,1) (0,2) (0,2) (0,1) (0,1)
Jugar
Crear
Unir
Apostar
Pasar
No ir
Igualar
Subir apuesta
Pedir carta
Descartar carta
(2,4) (1,1) (2,4) (0,2) (0,2) (0,2) (0,2) (0,2) (0,1) (0,1)
Administrar
Partida
Ganador
ID_Partida
Finalizado
Bote
Baraja
Claves
Jugadores
maxApuesta
Iniciado

6 Diseño

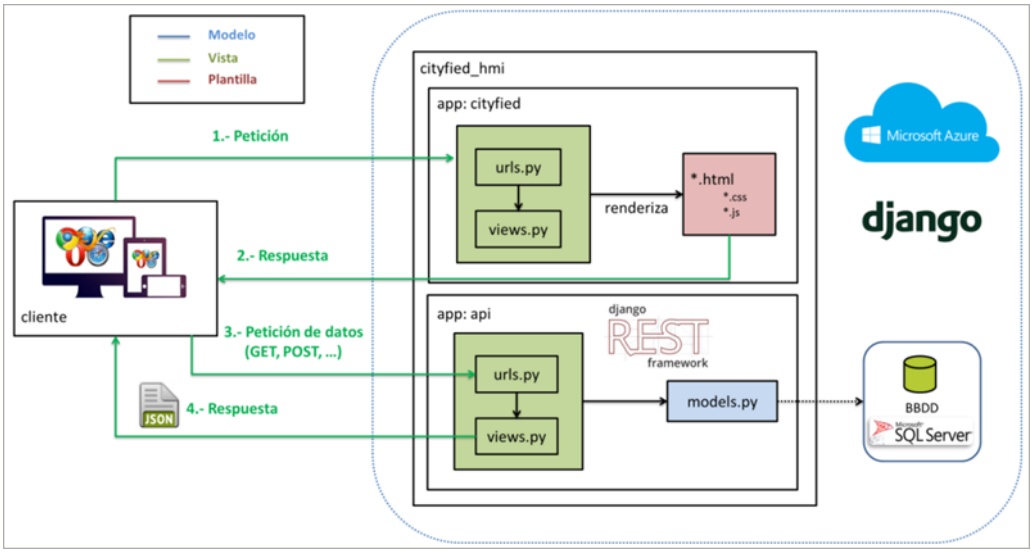
En este apartado de diseño se incluyen, entre otras cosas, los diagramas de arquitectura lógica y física, los cuales son de vital importancia en el ámbito de las aplicaciones empresariales.

6.1 Arquitectura lógica

La arquitectura lógica define las diferentes partes en las que se estructura una aplicación.

Para modelar la arquitectura de nuestra aplicación Jpoker se ha seguido un estilo arquitectónico por capas, el cual diferencia 3 capas principales:

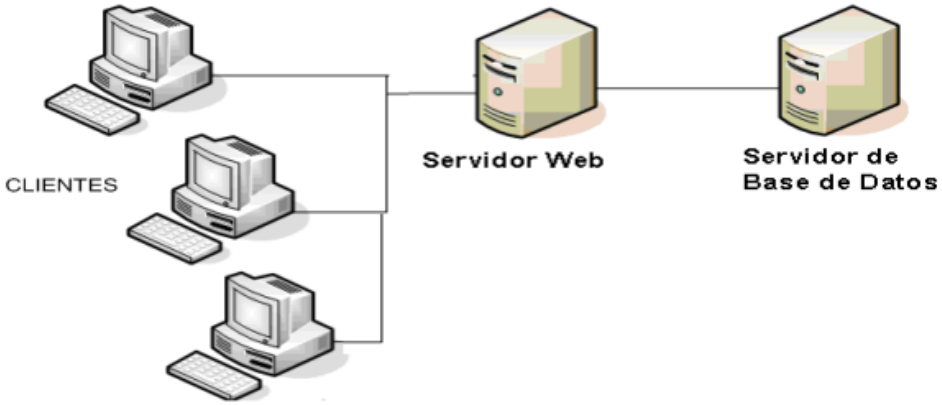
- Capa de Presentación
- Capa de Lógica de Negocio
- Capa de Acceso a Datos



Más adelante en el documento se explicará más del patrón MTV (Model Templete View) que utiliza Django.

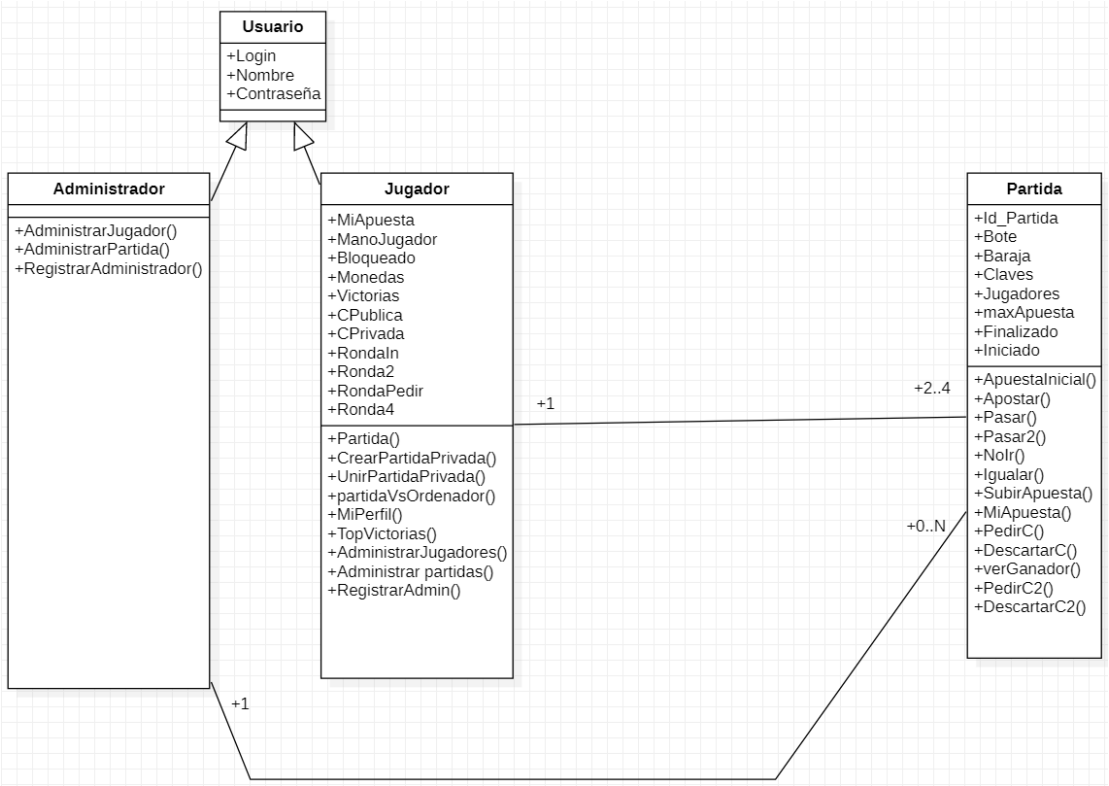
6.2 Arquitectura física

La arquitectura física identifica el ambiente físico de implantación de un sistema y tiene muy en cuenta las restricciones impuestas por los requisitos no funcionales que debe cumplir, de cara a utilizar unos componentes hardware u otros.



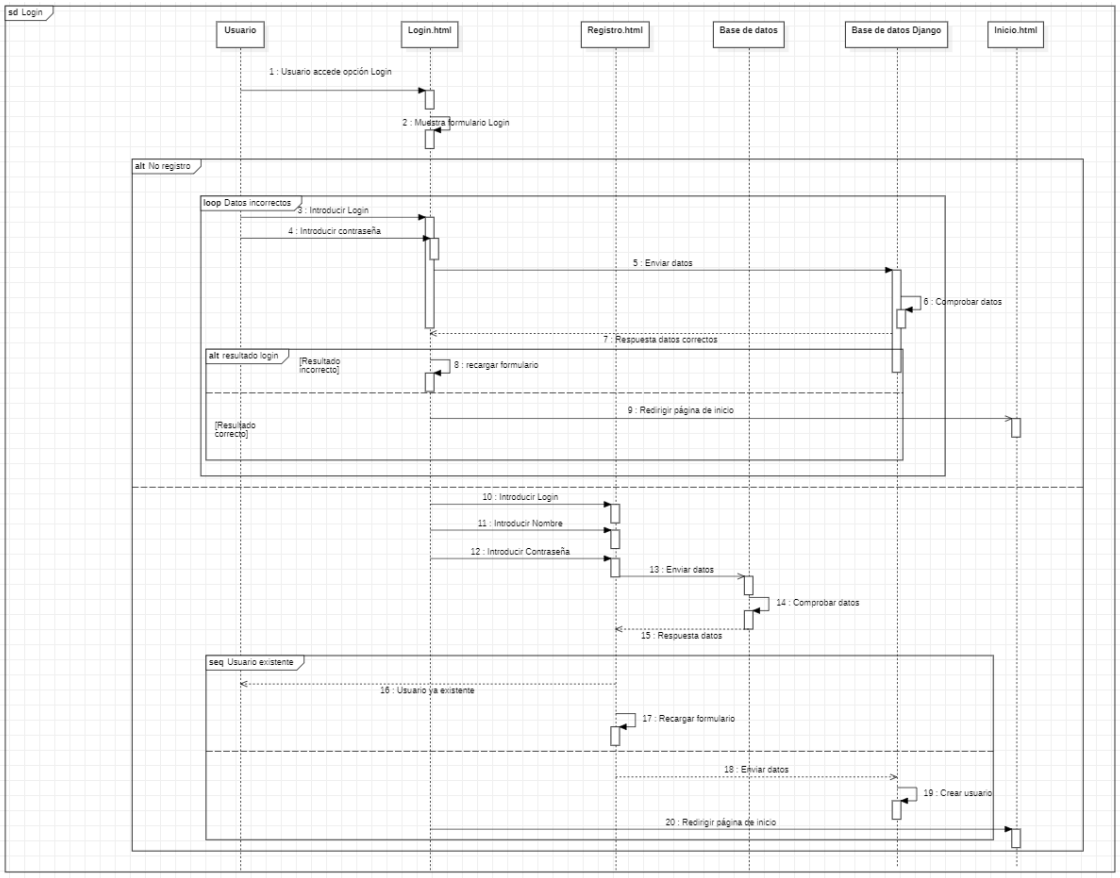
6.3 Diagrama de Clases Simplificado

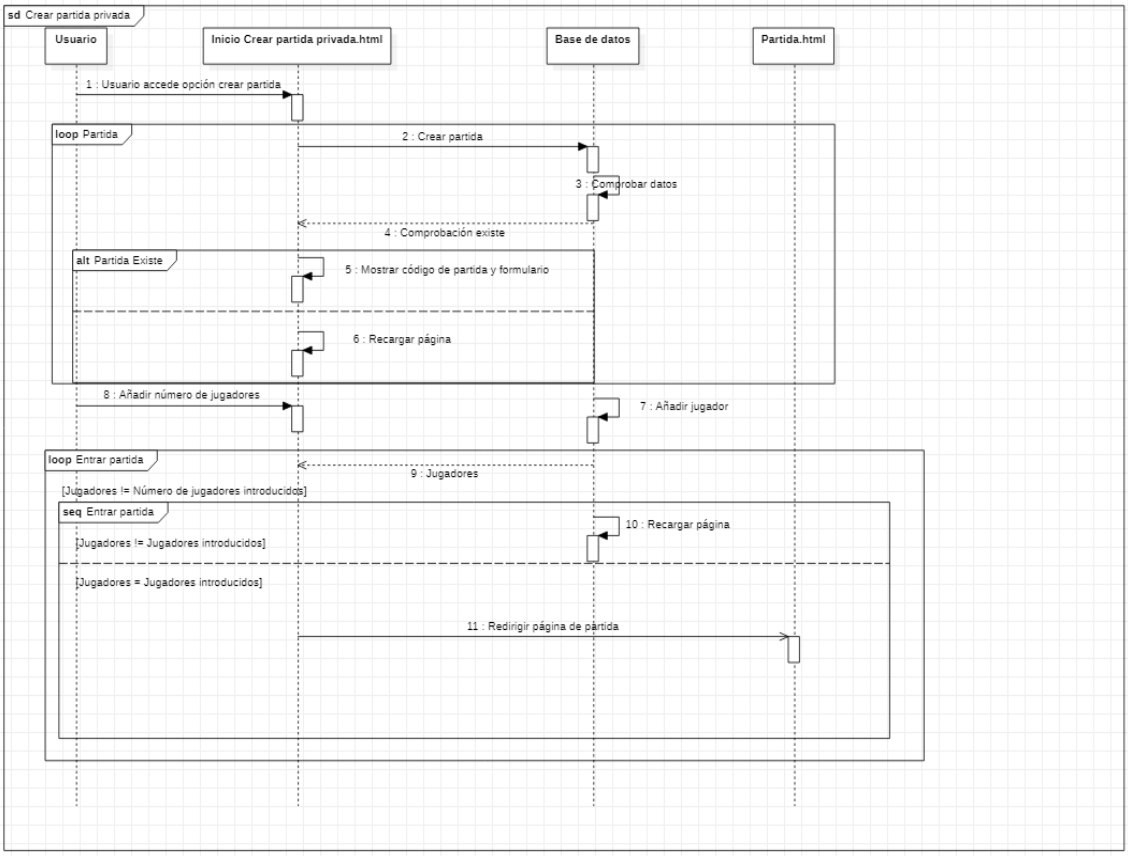
Este diagrama describe la estructura de un sistema en cuanto a las diferentes clases que lo componen y las relaciones que existen en ellas.

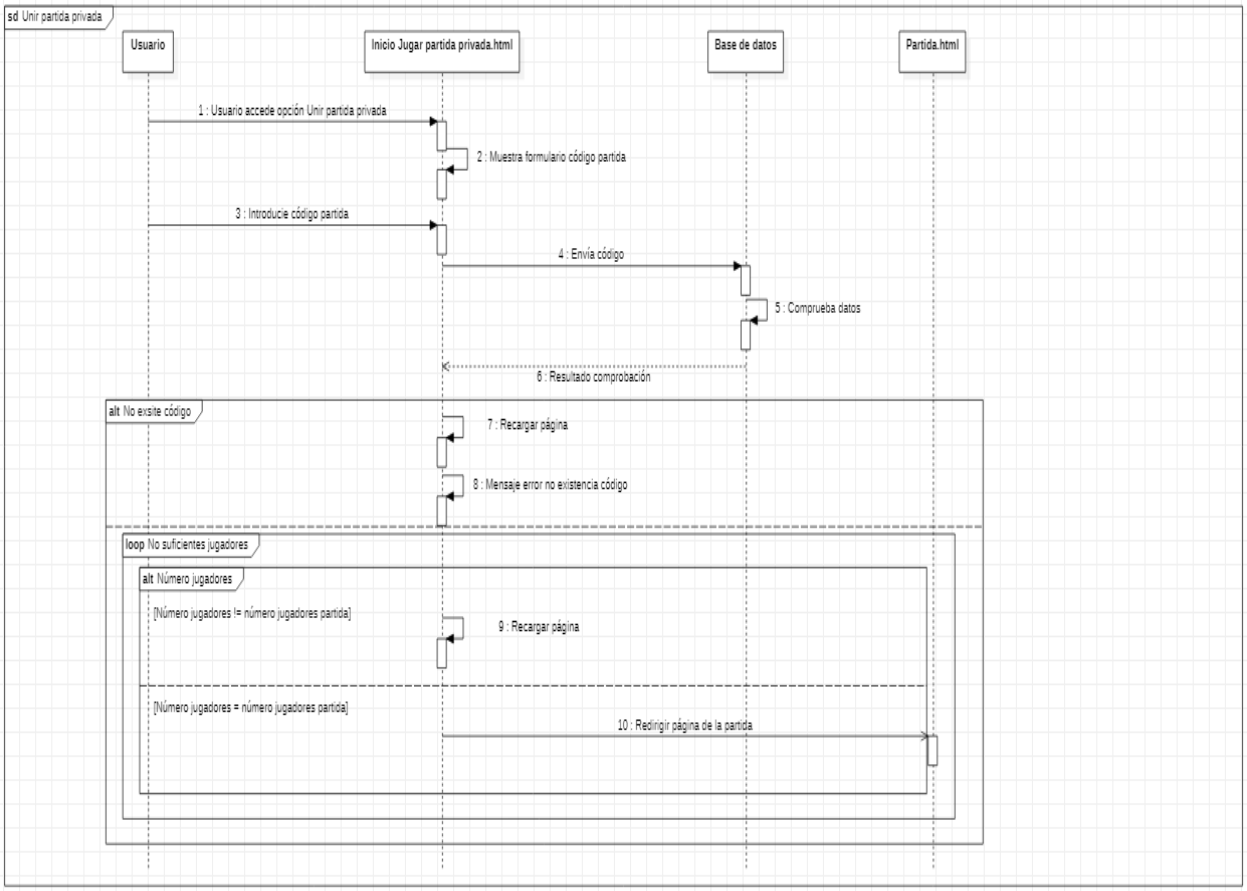


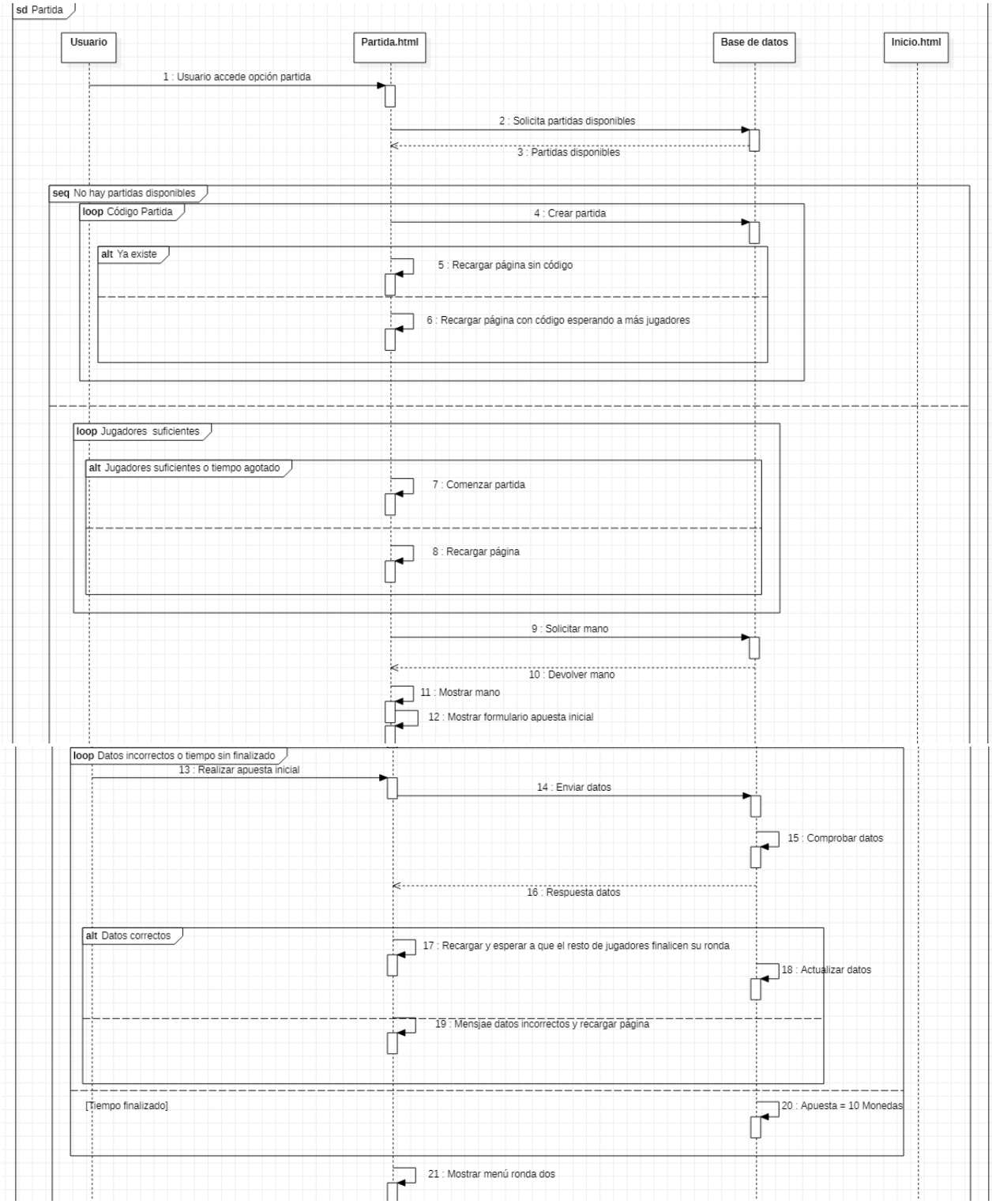
6.4 Diagramas de Secuencia

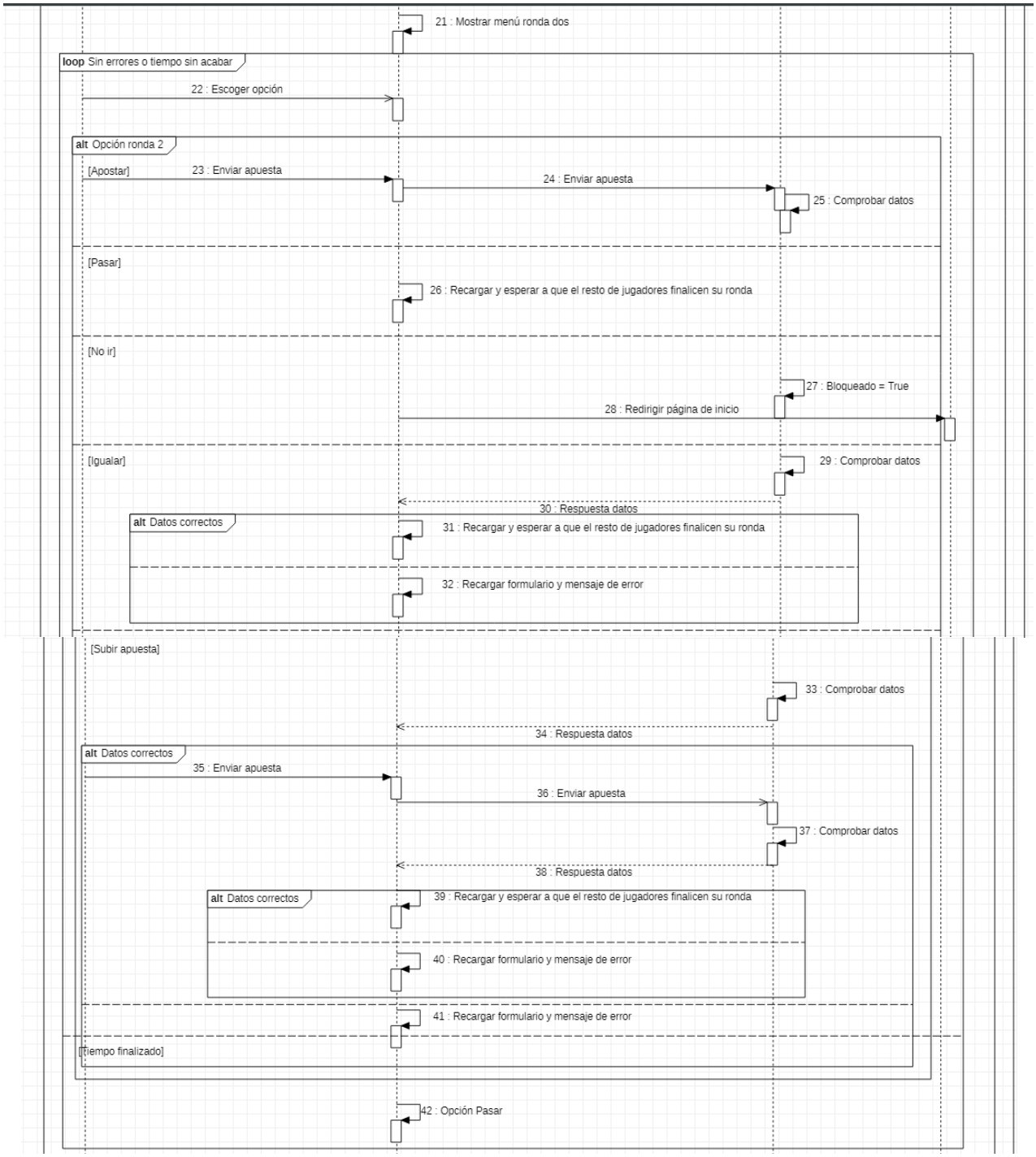
Estos diagramas permiten modelar las interacciones que se producen entre los distintos componentes de la aplicación con el fin de llevar a cabo una determinada tarea.

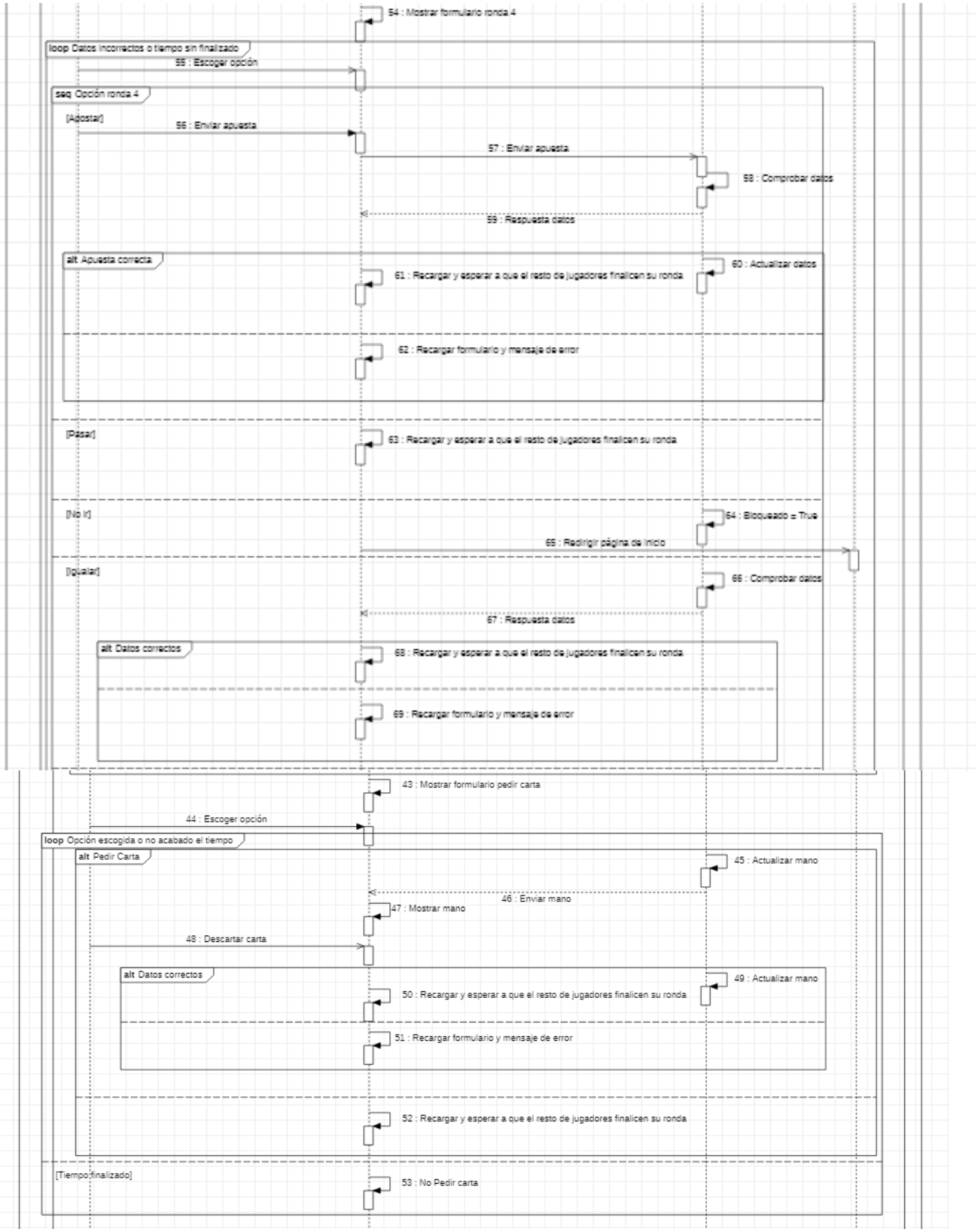


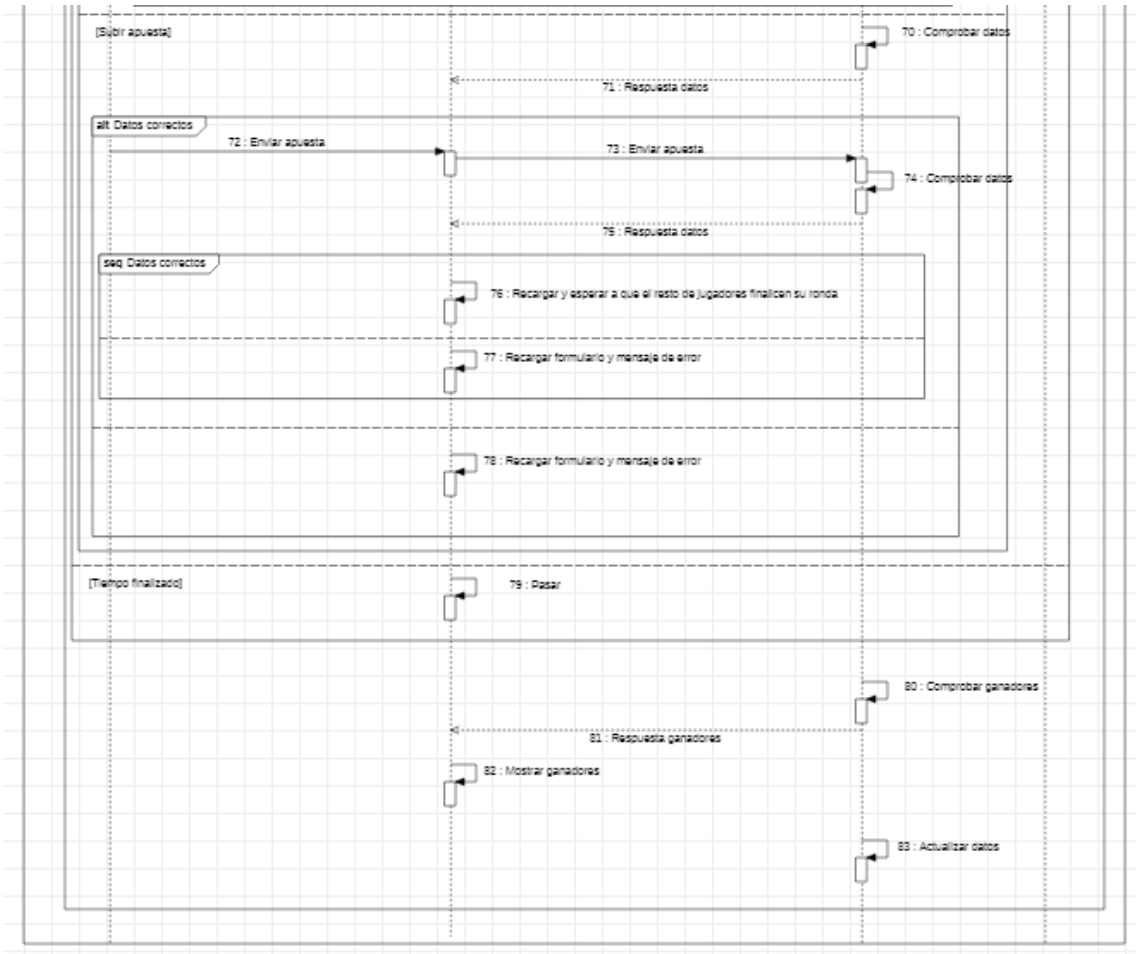

sd Crear partida privada
Usuario
Inicio Crear partida privada.html
Base de datos
Partida.html
1 : Usuario accede opción crear partida
loop Partida
2 : Crear partida
3 : Comprobar datos
4 : Comprobación existe
alt Partida Existe
5 : Mostrar código de partida y formulario
6 : Recargar página
8 : Añadir número de jugadores
7 : Añadir jugador
loop Entrar partida
9 : Jugadores
[Jugadores != Número de jugadores introducidos]
seq Entrar partida
10 : Recargar página
[Jugadores != Jugadores introducidos]
[Jugadores = Jugadores introducidos]
11 : Redirigir página de partida

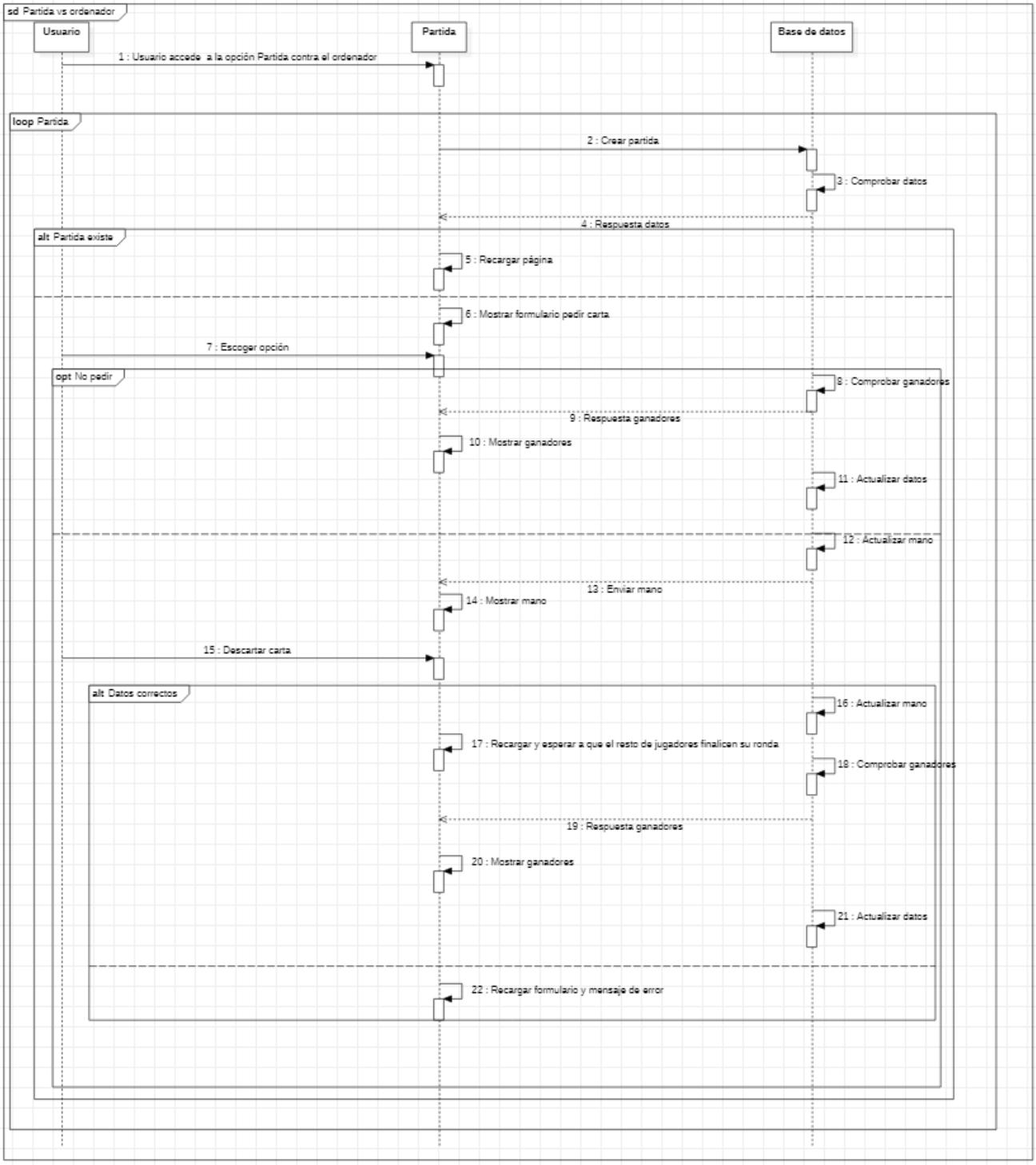

sd Unir partida privada
Usuario
Inicio Jugar partida privada.html
Base de datos
Partida.html
1 : Usuario accede opción Unir partida privada
2 : Muestra formulario código partida
3 : Introducie código partida
4 : Envía código
5 : Comprueba datos
6 : Resultado comprobación
alt No exsite código
7 : Recargar página
8 : Mensaje error no existencia código
loop No suficientes jugadores
alt Número jugadores
[Número jugadores != número jugadores partida]
9 : Recargar página
[Número jugadores = número jugadores partida]
10 : Redirigir página de la partida


sd Partida
Usuario
Partida.html
Base de datos
Inicio.html
1 : Usuario accede opción partida
2 : Solicita partidas disponibles
3 : Partidas disponibles
seq No hay partidas disponibles
loop Código Partida
4 : Crear partida
alt Ya existe
5 : Recargar página sin código
6 : Recargar página con código esperando a más jugadores
loop Jugadores suficientes
alt Jugadores suficientes o tiempo agotado
7 : Comenzar partida
8 : Recargar página
9 : Solicitar mano
10 : Devolver mano
11 : Mostrar mano
12 : Mostrar formulario apuesta inicial
loop Datos incorrectos o tiempo sin finalizado
13 : Realizar apuesta inicial
14 : Enviar datos
15 : Comprobar datos
16 : Respuesta datos
alt Datos correctos
17 : Recargar y esperar a que el resto de jugadores finalicen su ronda
18 : Actualizar datos
19 : Mensjae datos incorrectos y recargar página
[Tiempo finalizado]
20 : Apuesta = 10 Monedas
21 : Mostrar menú ronda dos


21 : Mostrar menú ronda dos
loop Sin errores o tiempo sin acabar
22 : Escoger opción
alt Opción ronda 2
[Apostar]
23 : Enviar apuesta
24 : Enviar apuesta
25 : Comprobar datos
[Pasar]
26 : Recargar y esperar a que el resto de jugadores finalicen su ronda
[No ir]
27 : Bloqueado = True
28 : Redirigir página de inicio
[Igualar]
29 : Comprobar datos
30 : Respuesta datos
alt Datos correctos
31 : Recargar y esperar a que el resto de jugadores finalicen su ronda
32 : Recargar formulario y mensaje de error
[Subir apuesta]
33 : Comprobar datos
34 : Respuesta datos
alt Datos correctos
35 : Enviar apuesta
36 : Enviar apuesta
37 : Comprobar datos
38 : Respuesta datos
alt Datos correctos
39 : Recargar y esperar a que el resto de jugadores finalicen su ronda
40 : Recargar formulario y mensaje de error
41 : Recargar formulario y mensaje de error
[Tiempo finalizado]
42 : Opción Pasar


54 : Mostrar formulario ronda 4
loop Datos incorrectos o tiempo sin finalizado
55 : Escoger opción
seq Opción ronda 4
[Apostar]
56 : Enviar apuesta
57 : Enviar apuesta
58 : Comprobar datos
59 : Respuesta datos
alt Apuesta correcta
60 : Actualizar datos
61 : Recargar y esperar a que el resto de jugadores finalicen su ronda
62 : Recargar formulario y mensaje de error
[Pasar]
63 : Recargar y esperar a que el resto de jugadores finalicen su ronda
[No ir]
64 : Bloqueado = True
65 : Redirigir página de inicio
[Igualar]
66 : Comprobar datos
67 : Respuesta datos
alt Datos correctos
68 : Recargar y esperar a que el resto de jugadores finalicen su ronda
69 : Recargar formulario y mensaje de error
43 : Mostrar formulario pedir carta
44 : Escoger opción
loop Opción escogida o no acabado el tiempo
alt Pedir Carta
45 : Actualizar mano
46 : Enviar mano
47 : Mostrar mano
48 : Descartar carta
alt Datos correctos
49 : Actualizar mano
50 : Recargar y esperar a que el resto de jugadores finalicen su ronda
51 : Recargar formulario y mensaje de error
52 : Recargar y esperar a que el resto de jugadores finalicen su ronda
[Tiempo finalizado]
53 : No Pedir carta


[Subir apuesta]
70 : Comprobar datos
71 : Respuesta datos
alt Datos correctos
72 : Enviar apuesta
73 : Enviar apuesta
74 : Comprobar datos
75 : Respuesta datos
seq Datos correctos
76 : Recargar y esperar a que el resto de jugadores finalicen su ronda
77 : Recargar formulario y mensaje de error
78 : Recargar formulario y mensaje de error
[Tiempo finalizado]
79 : Pasar
80 : Comprobar ganadores
81 : Respuesta ganadores
82 : Mostrar ganadores
83 : Actualizar datos


sd Partida vs ordenador
Usuario
Partida
Base de datos
1 : Usuario accede a la opción Partida contra el ordenador
loop Partida
2 : Crear partida
3 : Comprobar datos
4 : Respuesta datos
alt Partida existe
5 : Recargar página
6 : Mostrar formulario pedir carta
7 : Escoger opción
opt No pedir
8 : Comprobar ganadores
9 : Respuesta ganadores
10 : Mostrar ganadores
11 : Actualizar datos
12 : Actualizar mano
13 : Enviar mano
14 : Mostrar mano
15 : Descartar carta
alt Datos correctos
16 : Actualizar mano
17 : Recargar y esperar a que el resto de jugadores finalicen su ronda
18 : Comprobar ganadores
19 : Respuesta ganadores
20 : Mostrar ganadores
21 : Actualizar datos
22 : Recargar formulario y mensaje de error

6.5 Modelo Lógico de Datos

En este caso, al utilizar una base de datos MySQL, el modelo lógico de datos deberá ajustarse al modelo relacional.

Un modelo relacional trata los requisitos de información como si fueran tablas.

USUARIO - JUGADOR (Login, Nombre, Contraseña, MiApuesta, ManoJugador, Bloqueado, Monedas, Victorias, Cpublica, CPrivada, RondaIn, Ronda2, RondaPedir, Ronda4)

USUARIO - ADMINISTRADOR (Login, Nombre, Contraseña)

PARTIDA (Id_Partida, Bote, Baraja, Claves, Jugadores, maxApuesta, Finalizado, Iniciado)

6.6 Diccionario de datos

El diccionario de datos es una herramienta utilizada para describir las entidades y relaciones que componen la base de datos.

ID	***Entidad***	***Id***	***Relación***
E-01	Usuario	R-01	Jugar
E-02	Administrador	R-02	Crear
E-03	Partida	R-03	Unir
E-04	Perfil	R-04	Apostar
E-05	Ranking Ganadores	R-05	Pasar
	Reglas	R-06	No ir
		R-07	Igualar
		R-08	Subir Apuesta
		R-09	Pedir Carta
		R-10	Descartar Carta
		R-11	Mostrar
		R-12	Ver
		R-13	Gestionar
		R-14	Administrar

6.6.1 Entidades

E-01	**Usuario**				
Descripción	Es la persona que juega las partidas				
Normas					
Comentarios					
Atributos					
Id	**Nombre**	**Descripción**	**Tipo**	**Único**	**Null**
A-01	Login	Identificador del usuario	Varchar	Si	No
A-02	Nombre	Nombre del usuario	Varchar	No	No
A-03	Contraseña	Contraseña del usuario	Varchar	No	No
A-04	MiApuesta	Apuesta realizada durante la partida	Integer	No	Si
A-05	ManoJugador	Mano del jugador durante la partida	Json	No	Si
A-06	Bloqueado	Atributo que muestra si un usuario da a la opción “No ir” y no puede continuar la partida	Boolean	No	No
A-07	Monedas	Cantidad de monedas para apostar del usuario	Integer	No	No
A-08	Victorias	Número de victorias del jugador	Integer	No	Si
A-09	Cpublica	Clave de cifrado pública del jugador, varía en cada partida	Varchar	No	Si
A-10	CPrivada	Clave de cifrado privada del jugador, varía en cada partida	Varchar	No	Si
A-11	RondaIn	Atributo que muestra que ha terminado la ronda inicial	Boolean	No	No
A-12	Ronda2	Atributo que muestra que ha terminado la ronda 2	Boolean	No	No
A-13	RondaPedir	Atributo que muestra que ha terminado la ronda de pedir y descartar carta	Boolean	No	No
A-14	Ronda4	Atributo que muestra que ha terminado la ronda 4	Boolean	No	No

E-02	**Administrador**				
Descripción	Es la persona que juega las partidas				
Normas					
Comentarios					
Atributos					
Id	**Nombre**	**Descripción**	**Tipo**	**Único**	**Null**
A-01	Login	Identificador del usuario	Varchar	Si	No
A-02	Nombre	Nombre del usuario	Varchar	No	No
A-03	Contraseña	Contraseña del usuario	Varchar	No	No

E-03	**Partida**				
Descripción	Conjunto de jugadas en las que se pierde o gana un juego				
Normas					
Comentarios					
Atributos					
Id	**Nombre**	**Descripción**	**Tipo**	**Único**	**Null**
A-01	Id_Partida	Identificador de la partida	Varchar	Si	No
A-02	Bote	Bote de la partida	Integer	No	Si
A-03	Baraja	Baraja de cartas utilizada durante la partida	Json	No	Si
A-04	Claves	Claves de cifrado utilizadas durante la partida	Json	No	Si
A-05	Jugadores	Usuarios que juegas la partida	Json	No	Si
A-06	maxApuesta	Apuesta máxima realizada durante la partida	Integer	No	Si
A-07	Finalizado	Atributo que muestra que ha finalizado la partida	Boolean	No	No
A-08	Iniciado	Atributo que muestra que ha iniciado la partida	Boolean	No	No

E-04	**Perfil**				
Descripción	Descripción de los atributos principales del usuario				
Normas					
Comentarios					
Atributos					
Id	**Nombre**	**Descripción**	**Tipo**	**Único**	**Null**

E-05	**Ranking Ganadores**				
Descripción	Muestra de los usuarios con más victorias				
Normas					
Comentarios					
Atributos					
Id	**Nombre**	**Descripción**	**Tipo**	**Único**	**Null**

E-06	Reglas				
Descripción	Muestra las reglas de las partidas				
Normas					
Comentarios					
Atributos					
Id	**Nombre**	**Descripción**	**Tipo**	**Único**	**Null**

6.6.2 Relaciones

R-01		Jugar	
Descripción		Tomar parte en un juego organizado	
Nota			
Id	**Entidad**	**Participación**	**Cardinalidad**
E-01	Usuario	0	1
E-03	Partida	1	4

R-02		Crear	
Descripción		Producir una partida	
Nota			
Id	**Entidad**	**Participación**	**Cardinalidad**
E-01	Usuario	1	1
E-03	Partida		

R-03		Unir	
Descripción		Juntar dos o más jugadores distintos en una partida	
Nota			
Id	**Entidad**	**Participación**	**Cardinalidad**
E-01	Usuario	1	1
E-03	Partida	1	4

R-04		Apostar	
Descripción		Exponer una cantidad de monedas para tomar parte en un juego	
Nota			
Id	**Entidad**	**Participación**	**Cardinalidad**
E-01	Usuario	0	
E-03	Partida	0	

R-05		Pasar	
Descripción		Continuar al siguiente turno	
Nota			
Id	**Entidad**	**Participación**	**Cardinalidad**
E-01	Usuario	0	2
E-03	Partida	0	4

R-06		**No ir**	
Descripción		Finalización de la partida del usuario	
Nota			
Id	**Entidad**	**Participación**	**Cardinalidad**
E-01	Usuario	0	1
E-03	Partida	0	2

R-07		**Igualar**	
Descripción		Igualar mi apuesta a la máxima realizada por los usuarios de la partida	
Nota			
Id	**Entidad**	**Participación**	**Cardinalidad**
E-01	Usuario	0	2
E-03	Partida	0	2

R-08		**Subir Apuesta**	
Descripción		Igualar mi apuesta a la máxima realizada por los usuarios de la partida y tras ello aumentar mi apuesta	
Nota			
Id	**Entidad**	**Participación**	**Cardinalidad**
E-01	Usuario	0	2
E-03	Partida	0	2

R-09		**Pedir Carta**	
Descripción		Solicitar una carta de la baraja para mejorar la mano del usuario	
Nota			
Id	**Entidad**	**Participación**	**Cardinalidad**
E-01	Usuario	0	
E-03	Partida	0	

R-10		**Descartar Carta**	
Descripción		Descarta una carta de mi mano de la partida tras haber solicitado una anteriormente	
Nota			
Id	**Entidad**	**Participación**	**Cardinalidad**
E-01	Usuario	0	1
E-03	Partida	0	1

R-11		**Mostrar**	
Descripción		Mostrar el ranking con los 10 usuarios con más partidas ganadas	
Nota			
Id	**Entidad**	**Participación**	**Cardinalidad**
E-01	Usuario	0	N
E-05	Ranking Ganadores	0	N

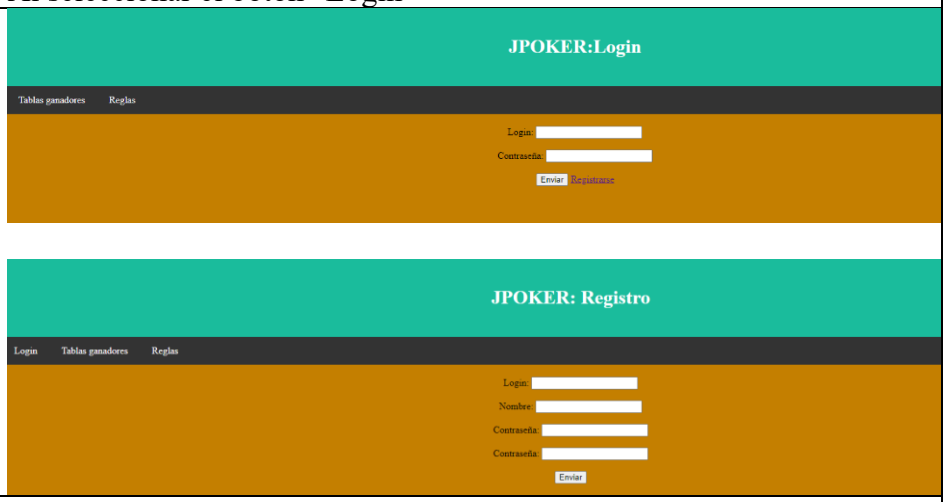
R-12		**Ver**	
Descripción		Observar los atributos principales del usuario	
Nota			
Id	**Entidad**	**Participación**	**Cardinalidad**
E-01	Usuario	0	N
E-06	Perfil	0	1

R-13		**Gestionar**	
Descripción		Gestión de usuarios	
Nota			
Id	**Entidad**	**Participación**	**Cardinalidad**
E-01	Usuario	1	N
E-02	Administrador	1	N

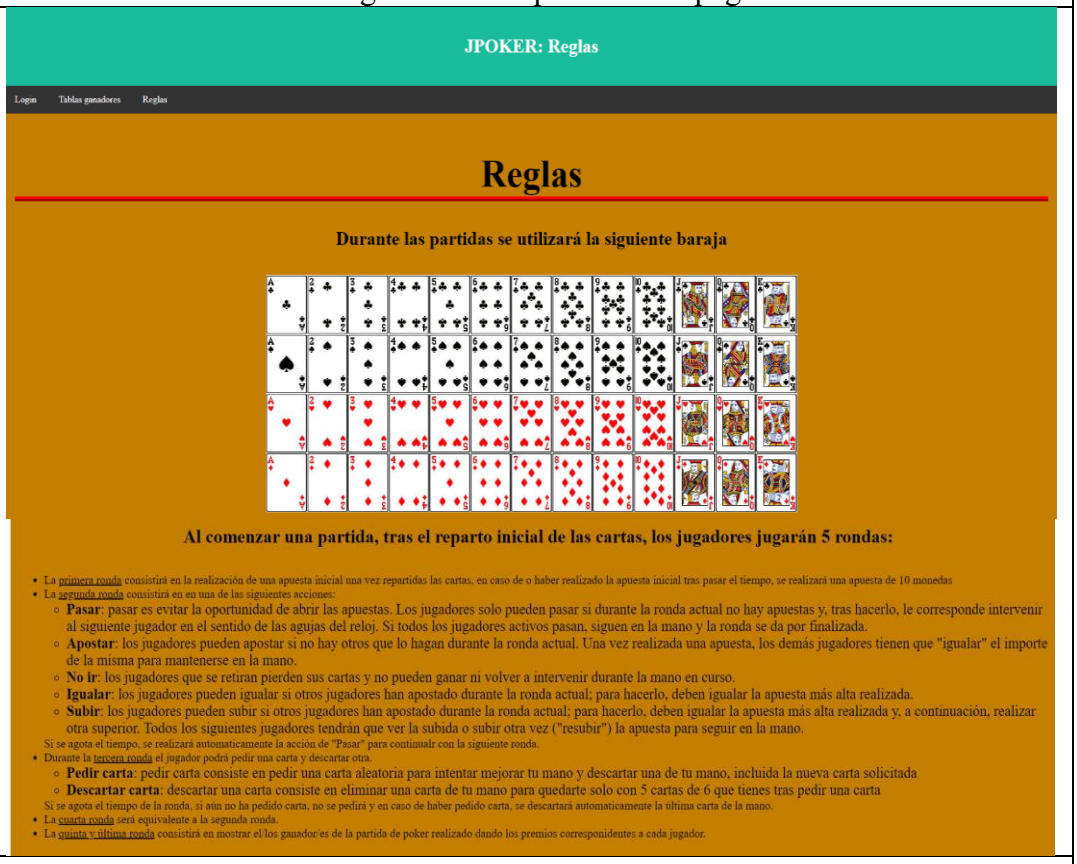
R-14		**Administrar**	
Descripción		Gestión de partidas	
Nota			
Id	**Entidad**	**Participación**	**Cardinalidad**
E-02	Administrador	1	N
E-03	Administrador	1	N

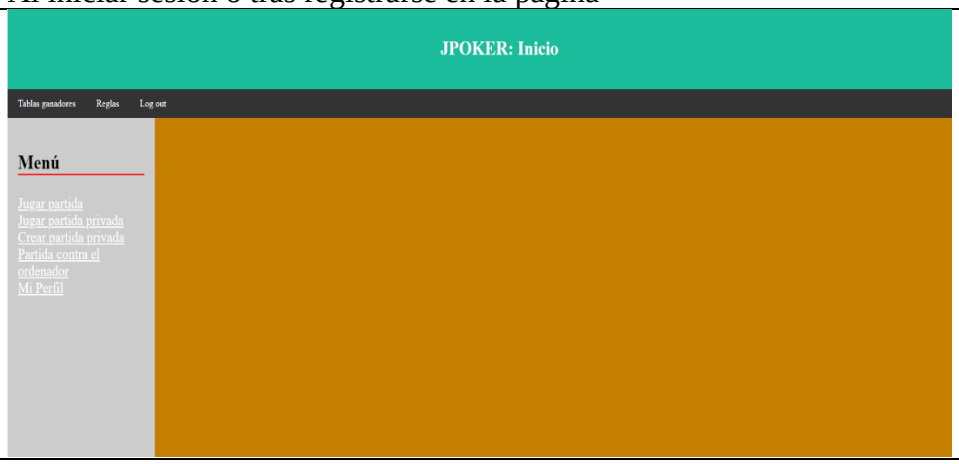
5.7 Diseño de Interfaz

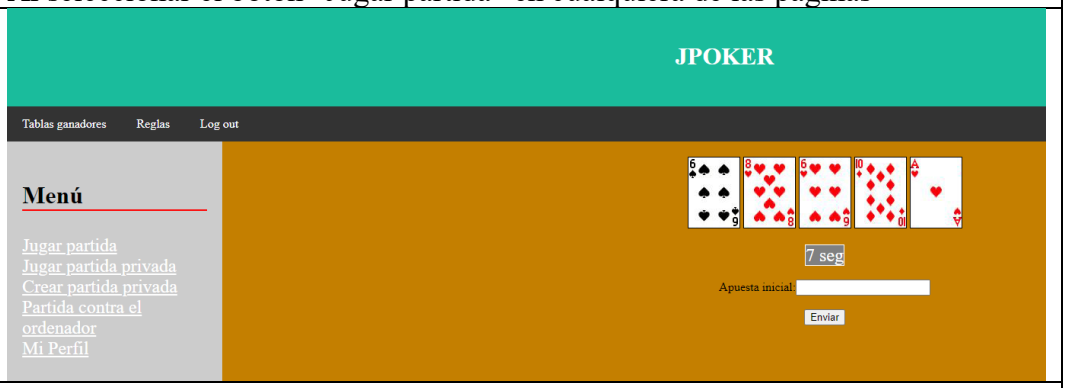
El diseño de la interfaz trata las consideraciones relacionadas con la interfaz de usuario que se le presentará a los usuarios de la aplicación.

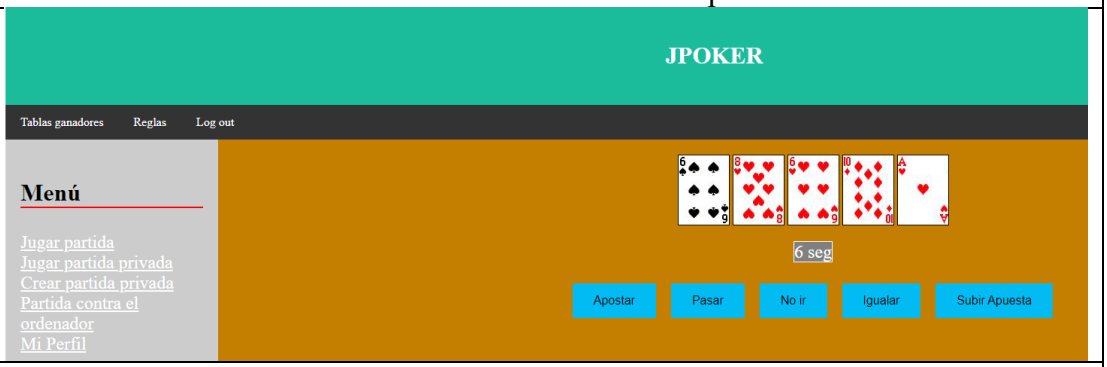
DI-01	Login
Descripción	Esta vista muestra como iniciar sesión en la aplicación
Activación	Al seleccionar el botón “Login”
Boceto	
Eventos	Inicia sesión y registrarse en la aplicación

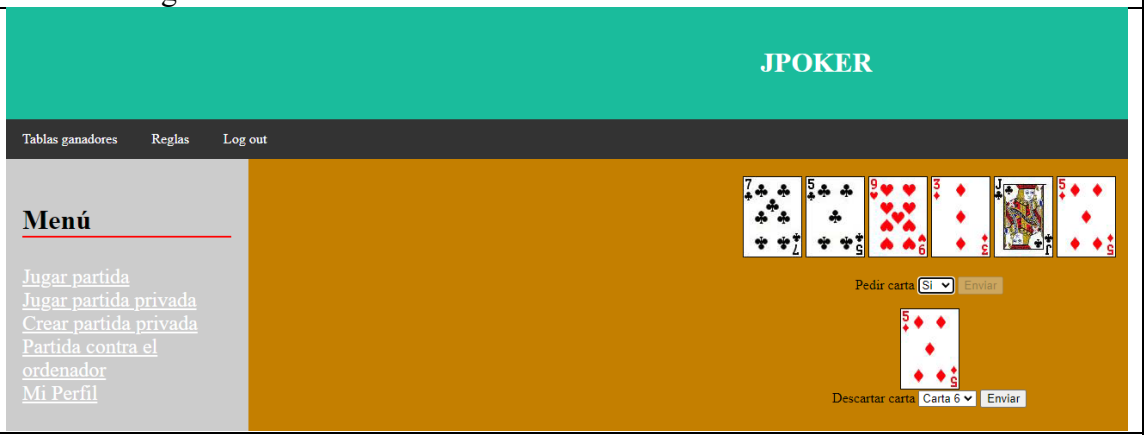
DI-02	Top Ganadores
Descripción	Esta vista muestra el ranking con los 10 usuarios con más partidas ganadas
Activación	Al seleccionar el botón “Tabla ganadores” en cualquiera de las páginas
Boceto	
Eventos	

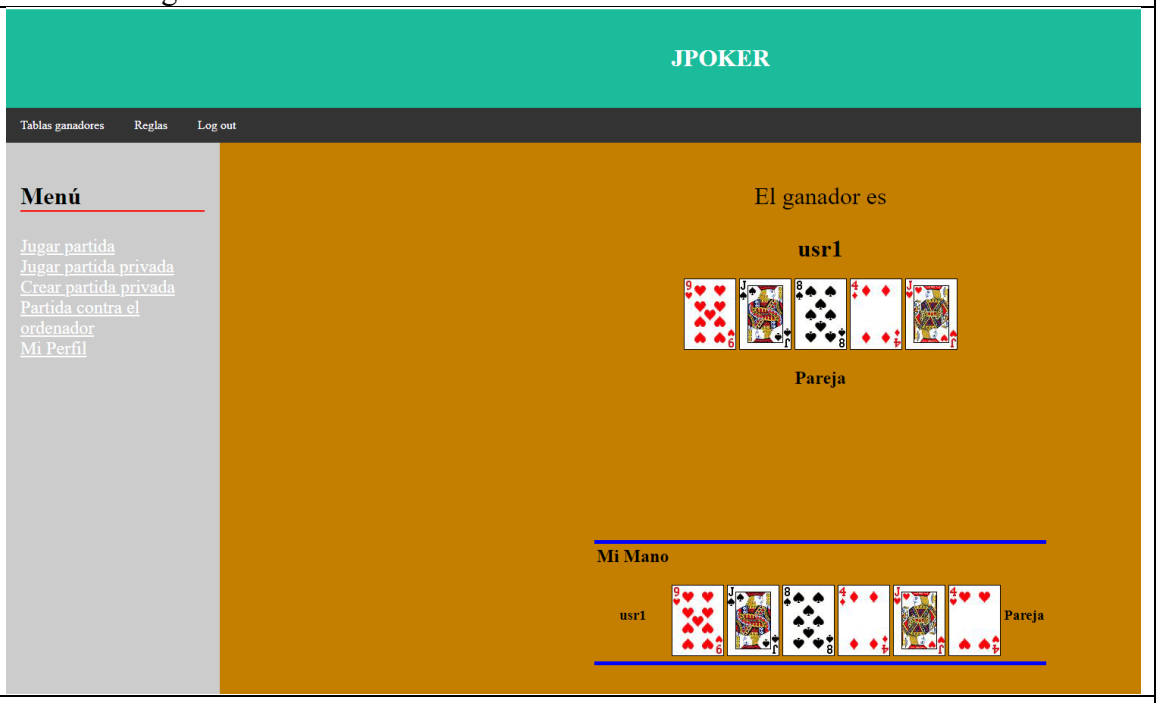
DI-03	Reglas
Descripción	Esta vista muestra las reglas de las partidas
Activación	Al seleccionar el botón “Reglas” en cualquiera de las páginas
Boceto	
Eventos	

DI-04	Inicio
Descripción	Esta vista muestra la página inicial tras iniciar sesión
Activación	Al iniciar sesión o tras registrarse en la página
Boceto	
Eventos	Jugar partida, crear partida privada, unirse a partida privada y ver perfil

DI-05	Apuesta inicial
Descripción	Esta vista muestra la página del primer movimiento tras iniciar partida
Activación	Al seleccionar el botón “Jugar partida” en cualquiera de las páginas
Boceto	
Eventos	Enviar la apuesta inicial de la partida

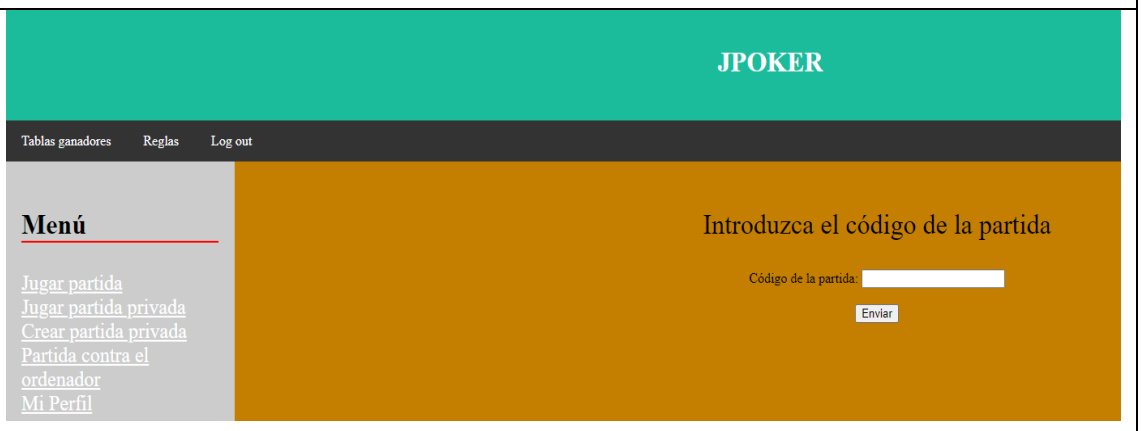
DI-06	Ronda 2 y ronda 4 de la partida
Descripción	Esta vista muestra la página del siguiente movimiento tras la apuesta inicial
Activación	Al seleccionar el botón “enviar” o al transcurrir el tiempo de la ronda
Boceto	
Eventos	Apostar, pasar, no ir, igualar o subir apuesta

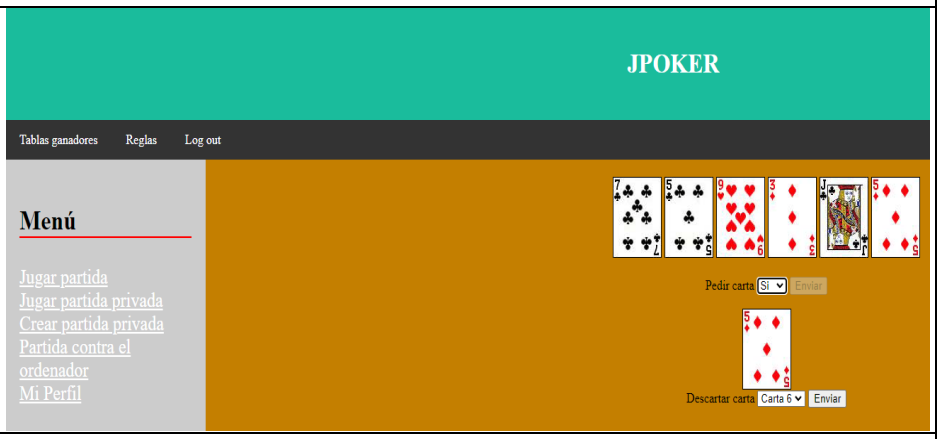
DI-07	Ronda pedir
Descripción	Esta vista muestra la página del siguiente movimiento tras la ronda 2
Activación	Al seleccionar el botón “enviar” apuesta, al transcurrir el tiempo de la ronda o al dar a “Pasar” o “Igualar”
Boceto	
Eventos	Pedir o no una carta y, en caso de pedir carta, descartar una de las cartas de la mano del usuario

DI-08	Ver ganadores
Descripción	Esta vista muestra los ganadores de la partida
Activación	Al seleccionar el botón “enviar” apuesta, al transcurrir el tiempo de la ronda o al dar a “Pasar” o “Igualar” en la ronda 4
Boceto	
Eventos	

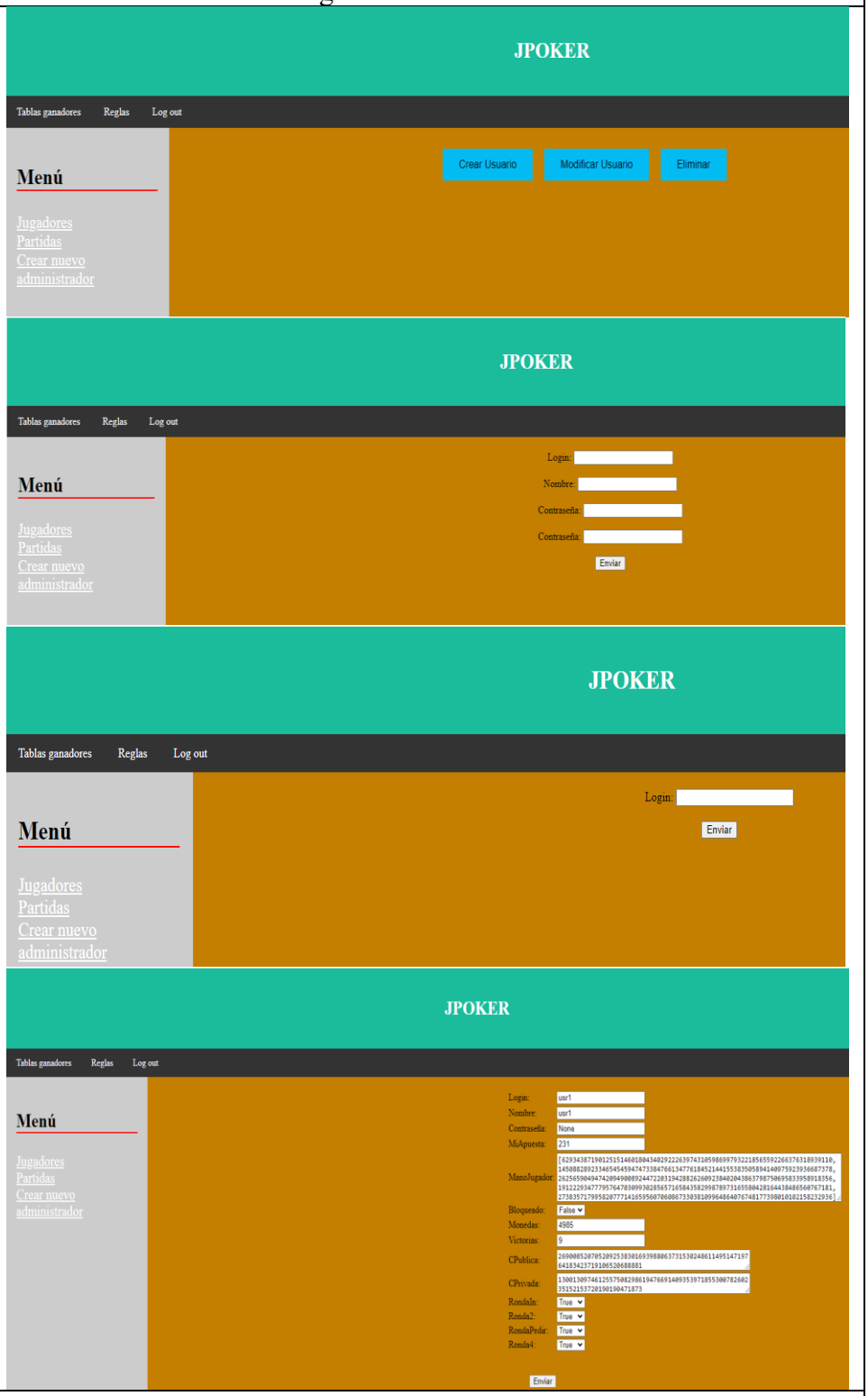
DI-09	Ver perfil
Descripción	Esta vista muestra el perfil de usuario
Activación	Al seleccionar el botón “Mi Perfil”
Boceto	
Eventos	

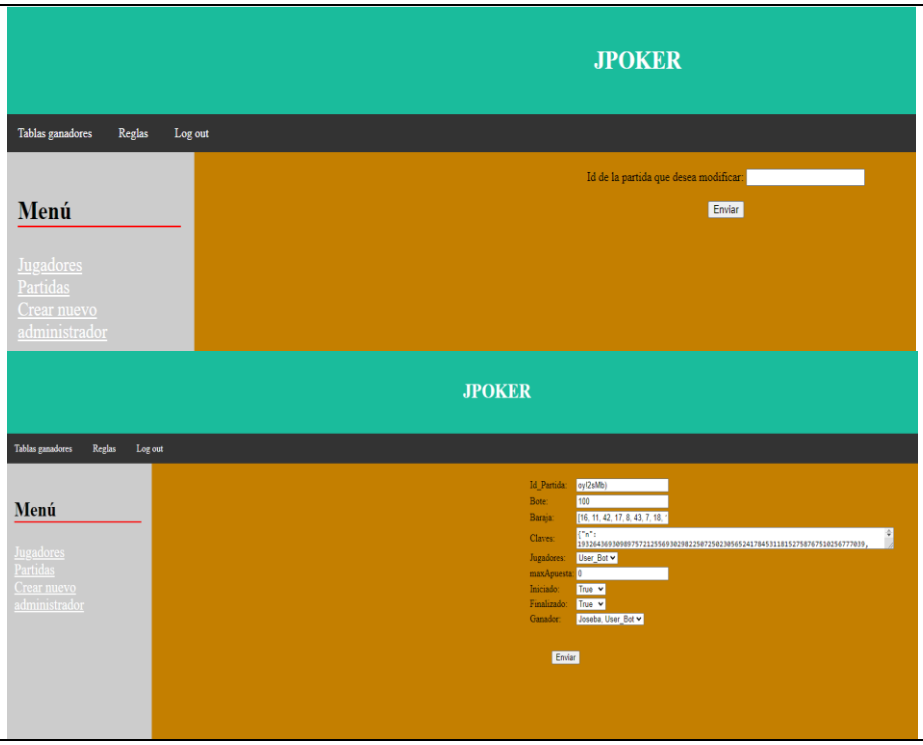
DI-10	Crear Partida
Descripción	Esta vista muestra cómo se crea una partida privada
Activación	Al seleccionar el botón “Crear partida privada”
Boceto	
Eventos	

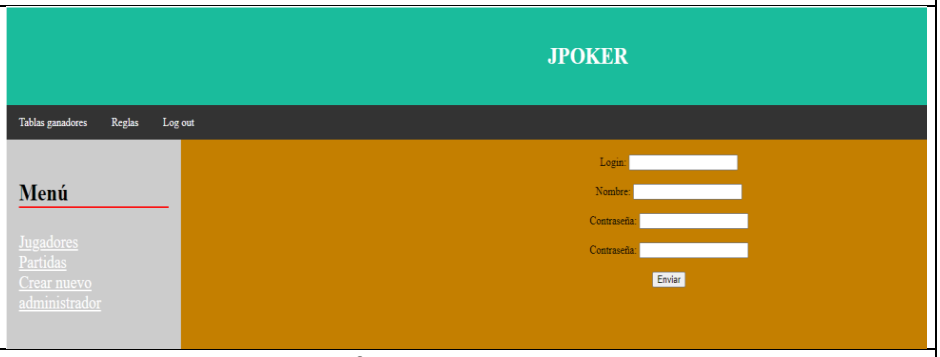
DI-11	Unir partida privada
Descripción	Esta vista muestra cómo se une a una partida privada
Activación	Al seleccionar el botón “Jugar partida privada”
Boceto	
Eventos	

DI-12	Partida contra el ordenador
Descripción	Esta vista muestra cómo se une a una partida contra el ordenador
Activación	Al seleccionar el botón “Partida contra el ordenador”
Boceto	
Eventos	Pedir carta o no, descartar carta en caso de pedir carta y ver los ganadores

DI-13	Inicio Administrador
Descripción	Esta vista muestra la página inicial tras iniciar sesión un administrador
Activación	Al iniciar sesión
Boceto	
Eventos	Administrar jugador, administrar partida y registrar nuevo jugador

DI-14	Administrar jugador
Descripción	Esta vista muestra la página de administración de los jugadores
Activación	Al seleccionar el botón “Jugadores”
Boceto	
Eventos	Crear nuevo jugador, modificar jugador, eliminar jugador

DI-15	Administrar partida
Descripción	Esta vista muestra la página de administración de las partidas
Activación	Al seleccionar el botón “Partidas”
Boceto	
Eventos	Introducir código de partida, modificación de los datos de la partida

DI-14	Registrar nuevo administrador
Descripción	Esta vista muestra la página de registro de nuevos administradores
Activación	Al seleccionar el botón “Crear nuevo administrador”
Boceto	
Eventos	Crear nuevo jugador, modificar jugador, eliminar jugador

7 Algoritmo de cifrado

Lo principal de este proyecto es el algoritmo de cifrado y protocolos criptográficos utilizados en los juegos de cartas online para dos o más jugadores, de forma de que el juego sea seguro y confidencial.
Para ello se ha implementado, en lenguaje Python, el algoritmo RSA.

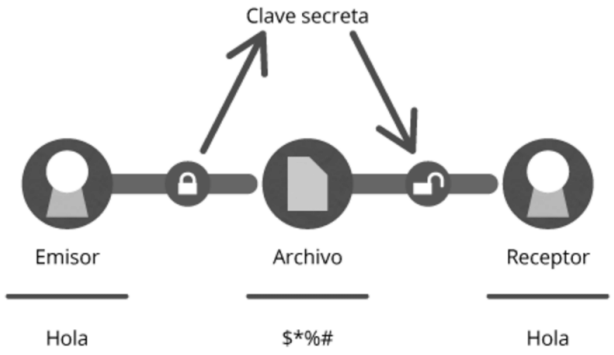
7.1 ¿Qué es la criptografía?

La criptografía es el desarrollo de un conjunto de técnicas que permiten alterar y modificar mensajes o archivos con el objetivo de que no puedan ser leídos por todos aquellos usuarios que no estén autorizados a hacerlo.

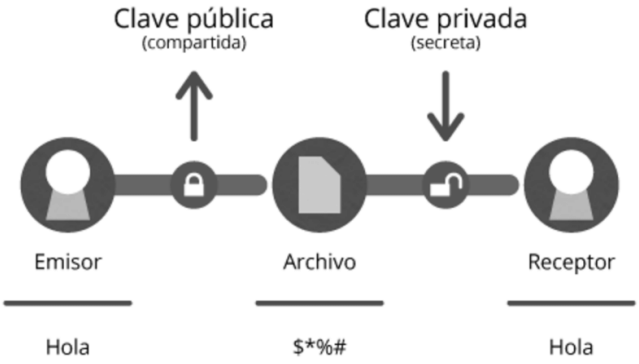
Denominamos proceso de **cifrado** cuando convertimos un mensaje que puede ser leído a uno que es ilegible, es decir que a un mensaje lo transformamos en caracteres que no construyen sentido. Y si queremos que, nuevamente, ese mensaje sea legible, hacemos el proceso inverso, al que denominamos **descifrado**. Por lo general estos mecanismos, que conforman la base de la criptografía, se realizan mediante algoritmos y claves.

En la actualidad existen diversos tipos de criptografía*:*

- **Criptografía simétrica**
  Es un método criptográfico que solo utiliza una clave para cifrar y descifrar los mensajes entre emisor y receptor. Debe ser conocidas por todas las partes involucradas.



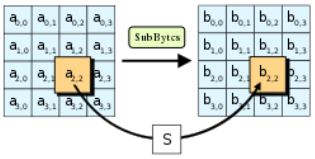
- **Criptografía asimétrica**
  Esta metodología tiene como base la utilización de dos claves diferentes, pero vinculadas matemáticamente entre sí, utilizadas para cifrar y descifrar el mensaje.
  Una de ellas debe ser pública, propia de cada participante, pero puesta a disposición de cualquier usuario.
  La otra es una clave privada, también propia de cada uno de ellos, pero que debe permanecer en secreto y nunca ser revelada.



7.2 Diferentes algoritmos de cifrado

Algunos de los algoritmos de cifrado más conocidos son los siguientes:

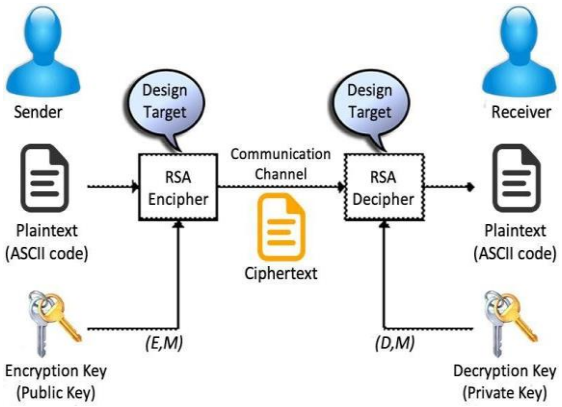
- **AES (Advanced Encryption Standard)**, algoritmo simétrico es un esquema de cifrado por bloques adoptado como un estándar de cifrado.
  Opera con bloques de 128 bits, y admite claves de 128, 192 y 256 bits (AES128, AES192 y AES256). En los tres casos, el algoritmo consta de una transformación inicial y un numero Nr de vueltas que depende de la longitud de la clave (Nr = 10 para 128 bits, Nr = 12 para 192 bits, y Nr = 14 para 256 bits). El resultado E obtenido en cada etapa intermedia del algoritmo se llama estado. Todas las vueltas son "regulares" menos la vuelta final, que es diferente.



- **DSA** (**Digital Signature Algorithm)**, algoritmo asimétrico. Este algoritmo sirve para firmar (autenticar), pero no para cifrar información.

- **RSA** ***(Rivest, Shamir y Adleman)***, algoritmo asimétrico. es un sistema criptográfico de clave pública desarrollado en 1979, que utiliza factorización de números enteros. Hablaremos más en profundidad más adelante.
- **SHA**. Los **algoritmos de hash seguro** son una familia de funciones de hash criptográficas publicadas como un estándar federal de procesamiento de información. Fue creado para ser usado junto al algoritmo DSA. El algoritmo actúa a partir de un mensaje de menos de 2^{64} bits, el SHA produce una salida de 160 bits, con un proceso similar al MD5

7.3 RSA



En criptografía, RSA *(Rivest, Shamir y Adleman)* es un sistema criptográfico de clave pública desarrollado en 1979, que utiliza factorización de números enteros. Es el primer y más utilizado algoritmo de este tipo y es válido tanto para cifrar como para firmar digitalmente.

La seguridad de este algoritmo radica en el problema de la factorización de números enteros. Los mensajes enviados se representan mediante números, y el funcionamiento se basa en el producto, conocido, de dos números primos grandes elegidos al azar y mantenidos en secreto.

Como en todo sistema de clave pública, cada usuario posee dos claves de cifrado: una pública y otra privada. Cuando se quiere enviar un mensaje, el emisor busca la clave pública del receptor, cifra su mensaje con esa clave, y una vez que el mensaje cifrado llega al receptor, este se ocupa de descifrarlo usando su clave privada.

Como se implementa:

1- Para obtener las claves hay obtenerlas siguiendo los siguientes pasos

I. Cada usuario i debe elegir una pareja de primos distintos pi, qi >> 0

II. Se calcula ni = pi * qi

III. Se calcula la funcion Phi de Euler

$\Phi(ni) := (pi - 1) * (qi - 1)$

IV. Se elige aleatoriamente $1 < ei < \Phi(ni)$ tal que $mcd(ei, \Phi(ni)) = 1$

V. Se calcula el inverso modular $di \equiv e{-}1\ i \pmod{\Phi(ni)}$

VI. $ei \cdot di \bmod \Phi(ni) = 1$

Por tanto, las claves son:

- Clave pública: (ni, ei)
- Clave privada: di

2- Cifrado

Siendo M el número que queremos cifrar:

$$M \rightarrow C \equiv M^{ei} \pmod{ni}$$

3- Descifrado

$$C \rightarrow C^{di} \equiv M^{ei*di} \pmod{ni}$$

Teorema:

$$M^{ei*di} \equiv M \pmod{ni}$$

Es decir, al descifrar el criptograma con la clave privada se recupera el mensaje original.

6.4 Como funciona en nuestra aplicación

Supongamos que estamos en una partida de 4 jugadores, Jugador A, Jugador B, Jugador C y Jugador D. A dispone de la baraja:

I. Jugador A cifra la baraja con su clave pública.
II. Jugador A le da la baraja a Jugador B.
III. Jugador B cifra la baraja con su clave pública y escoge 5 cartas.
IV. Jugador A descifra la baraja y las 5 cartas escogidas con su clave privada y se la devuelve al Jugador B.
V. Jugador B almacena las 5 cartas en su mano y pasa la baraja al Jugador C.
VI. Jugador C cifra la baraja con su clave pública y escoge 5 cartas.
VII. Jugador B descifra la baraja y las 5 cartas escogidas con su clave privada y se la devuelve al Jugador C.
VIII. Jugador C almacena las 5 cartas en su mano y pasa la baraja al Jugador D.
IX. Jugador D cifra la baraja con su clave pública y escoge 5 cartas.
X. Jugador C descifra la baraja y las 5 cartas escogidas con su clave privada y se la devuelve al Jugador D.
XI. Jugador D almacena las 5 cartas en su mano y pasa la baraja al Jugador A.
XII. Jugador A cifra la baraja con su clave pública y escoge 5 cartas.
XIII. Jugador D descifra la baraja y las 5 cartas escogidas con su clave privada y se la devuelve al Jugador A.
XIV. Jugador D almacena las 5 cartas en su mano y pasa la baraja al Jugador A.

Esto ocurre durante el reparto inicial, pero durante la partida, cuando pides carta, ocurre de la siguiente forma:

- Si solicita carta cualquier jugador diferente del jugador A:
  I. Jugador A cifra la baraja con su clave pública.
  II. Jugador A le da la baraja a Jugador B.
  III. Jugador B cifra la baraja con su clave pública y escoge 1 carta.
  IV. Jugador A descifra la baraja y la carta escogida con su clave privada y se la devuelve al Jugador B.
  V. Jugador B almacena la carta en su mano.
  VI. Jugador B descifra la baraja con su clave privada.

- Si solicita carta el jugador A:
  I. Jugador D cifra la baraja con su clave pública.
  II. Jugador D le da la baraja a Jugador A.
  III. Jugador A cifra la baraja con su clave pública y escoge 1 carta.
  IV. Jugador D descifra la baraja y la carta escogida con su clave privada y se la devuelve al Jugador A.
  V. Jugador A almacena la carta en su mano.
  VI. Jugador A descifra la baraja con su clave privada.

7.5 Seguridad y confidencialidad en el juego

La criptografía actual permite proteger la información contra accesos no autorizados, lo que garantiza su confidencialidad a la vez que provee mecanismos para asegurarla.

Un buen sistema de cifrado pone toda la seguridad en la clave y ninguna en el algoritmo. Es decir, que no debería favorecer conocer el algoritmo utilizado. El algoritmo AES, citado anteriormente, posee esta propiedad.

Dado que la seguridad radica en la clave, es importante que no sea conocido el tipo de clave y cuanto mayor sea el espacio de posibilidades más seguro será.

El algoritmo utilizado en esta plataforma web es el RSA, que, para crear la claves, se cogen dos números primos grandes, en este caso los números primos se escogen aleatoriamente en un rango entre 2^{100-25} y 2^{100+25} para que sean suficientemente grandes proporcionando así mayor seguridad dentro de la aplicación.

En esta plataforma web es importante el uso de la criptografía, estamos ante un juego de cartas en el que es importante que los jugadores rivales no conozcan cuáles son tus cartas para poder ganar la partida, del mismo modo que es importante no conocer las cartas del resto de jugadores para obtener una gran ventaja respecto del resto.

Las claves se generan de nuevo en cada partida ya que sería muy inseguro mantener las mismas siempre porque sería necesario mantener los números primos necesarios para crear la clave, tarea que facilitaría al programador que atacase la aplicación. Además, cada jugador genera sus propias claves, esto permite implementar la seguridad del sistema.

8 Pruebas

8.1 Prueba de Caja Negra

Las pruebas de caja negra son utilizadas para validar la corrección de las salidas que proporciona el sistema a partir de unos datos de entrada establecidos.

Para la realización de estas últimas pruebas hay creado un administrador con los siguientes datos:

- Jugador
  - **Login**: Usuario
  - **Contraseña**: Usuario
- Administrador
  - **Login**: Administrador
  - **Contraseña**: Administrador

Nota: Al entrar a la página inicial (http://127.0.0.1:8000/poker/) se crearán ambos usuarios.

Nota: Para realizar las pruebas de partidas se deberán hacer en distintos navegadores para evitar la concurrencia de sesiones.

PCN-01	**Loguearse en la aplicación**
Propósito	Autenticar la identidad de un usuario registrado en la aplicación.
Prerrequisitos	El usuario debe estar registrado en la aplicación
Datos de entrada	Login y contraseña correctos
Resultado esperado	El usuario accede a la aplicación
Resultado obtenido	El usuario accede a la aplicación por medio de su página privada de inicio

PCN-02	**Jugar partida**
Propósito	Jugar una partida de poker contra otros jugadores
Prerrequisitos	El usuario debe estar logueado y haber accedido al botón de “Jugar partida”
Datos de entrada	
Resultado esperado	Jugar y ganar una partida contra otros jugadores con sus rondas correspondientes
Resultado obtenido	Jugar una partida contra otros jugadores con sus rondas correspondientes

PCN-03	**Crear partida privada**
Propósito	Crear una partida de poker privada
Prerrequisitos	El usuario debe estar logueado y haber accedido al botón de "Crear partida privada"
Datos de entrada	Número de jugadores de la partida (hasta 4)
Resultado esperado	Crear una partida de poker
Resultado obtenido	Partida de poker creada

PCN-04	**Unir partida privada**
Propósito	Unirse una partida de poker privada
Prerrequisitos	El usuario debe estar logueado y haber accedido al botón de "Jugar partida privada"
Datos de entrada	Código de la partida
Resultado esperado	Unirse a una partida de poker privada
Resultado obtenido	Unión a una partida de poker privada

PCN-05	**Jugar partida contra el ordenador**
Propósito	Jugar una partida de poker contra el ordenador
Prerrequisitos	El usuario debe estar logueado y haber accedido al botón de "Partida contra el ordenador"
Datos de entrada	
Resultado esperado	Jugar y ganar una partida el ordenador con sus rondas correspondientes
Resultado obtenido	Jugar una partida contra el ordenador con sus rondas correspondientes

PCN-06	**Ver mi perfil de usuario**
Propósito	Mostrar el perfil de usuario con algunos datos como el número de monedas y en número de victorias conseguidas
Prerrequisitos	El usuario debe estar logueado en la aplicación
Datos de entrada	
Resultado esperado	El usuario accede a su perfil
Resultado obtenido	El usuario ve a su perfil con sus datos

PCN-07	**Ver reglas**
Propósito	Ver las reglas del poker utilizadas en la aplicación
Prerrequisitos	
Datos de entrada	
Resultado esperado	Mostrar las reglas del poker
Resultado obtenido	Mostrar las reglas del poker

PCN-08	**Ver tabla usuarios con más victorias**
Propósito	Ver una tabla con el top 10 de usuarios de la aplicación con más victorias
Prerrequisitos	
Datos de entrada	
Resultado esperado	Mostrar la tabla con el ranking
Resultado obtenido	Mostrar la tabla con el ranking

PCN-09	**Administrar jugadores**
Propósito	Crear, modificar o eliminar un jugador
Prerrequisitos	El usuario debe estar logueado y haber accedido al botón de “Jugadores”
Datos de entrada	
Resultado esperado	Creación, modificación o eliminación de un jugador.
Resultado obtenido	Creación, modificación o eliminación de un jugador.

PCN-10	**Administrar partidas**
Propósito	Modificar los datos de una partida
Prerrequisitos	El usuario debe estar logueado y haber accedido al botón de “Partidas”
Datos de entrada	Identificador de la partida
Resultado esperado	Modificación de los datos de la partida
Resultado obtenido	Modificación de los datos de la partida

PCN-11	**Registrar nuevos administradores**
Propósito	Agregar un nuevo administrador
Prerrequisitos	El usuario debe estar logueado y haber accedido al botón de “Crear Nuevo Administrador”
Datos de entrada	Login, nombre y contraseña
Resultado esperado	Creación de un nuevo administrador
Resultado obtenido	Creación de un nuevo administrador

8.2 Prueba de Caja Blanca

Las pruebas de caja blanca son utilizadas para validar las funciones internas de los módulos o subprogramas.

PCB-01	**Usuario bloqueado**
Propósito	Mostrar que no permite seguir jugando la partida
Prerrequisitos	Jugar una partida y dar la opción del menú "No ir"
Datos de entrada	Dar al botón de retroceso para intentar continuar jugando la partida
Resultado esperado	No permitir seguir jugando la partida
Resultado obtenido	Ir a la pantalla de inicio y no permitir volver a la página anterior

PCB-02	**Partida no iniciada**
Propósito	Mostrar que no permite seguir jugando la partida porque aún no ha comenzado la partida
Prerrequisitos	Intentar jugar una ronda de la partida sin haber comenzado la misma
Datos de entrada	
Resultado esperado	No permitir jugar la partida
Resultado obtenido	Ir a la pantalla de inicio y no permitir jugar la partida

PCB-03	**Partida finalizada**
Propósito	Mostrar que no permite seguir jugando la partida porque la partida ha finalizado
Prerrequisitos	Intentar jugar una ronda de la partida que ya ha finalizado
Datos de entrada	
Resultado esperado	No permitir jugar la partida
Resultado obtenido	Ir a la pantalla de inicio y no permitir jugar la partida

PCB-04	Igualar apuesta
Propósito	Mostrar que no se permite igualar una apuesta si no tiene monedas suficientes
Prerrequisitos	Tener menos monedas que la apuesta más grande
Datos de entrada	
Resultado esperado	Si no tiene monedas suficientes recargará la página, añadirá mensaje de error y permitirá hacer cualquier otra opción
Resultado obtenido	Se recarga la página, añade mensaje de error y permite hacer cualquier otra opción de la ronda

PCB-05	Subir apuesta
Propósito	Mostrar que no se permite subir una apuesta si no tiene monedas suficientes
Prerrequisitos	Tener menos monedas que la apuesta más grande o no tener más monedas para apostar
Datos de entrada	
Resultado esperado	Si no tiene monedas suficientes recargará la página, añadirá mensaje de error y permitirá hacer cualquier otra opción
Resultado obtenido	Se recarga la página, añade mensaje de error y permitirá hacer cualquier otra opción de la ronda

PCB-06	Apostar
Propósito	Mostrar que no se permite apostar si no tiene monedas suficientes
Prerrequisitos	No tener tantas monedas como se han apostado
Datos de entrada	
Resultado esperado	Si no tiene monedas suficientes recargará la página, añadirá mensaje de error y permitirá hacer cualquier otra opción de la ronda
Resultado obtenido	Se recarga la página, añade mensaje de error y permite hacer cualquier otra opción de la ronda

PCB-07	**Apostar**
Propósito	No poder apostar más monedas de las que tienes
Prerrequisitos	Iniciar una partida, ya sea pública o privada y hacer la apuesta inicial o la opción de apostar en la segunda o cuarta ronda
Datos de entrada	
Resultado esperado	Se recargará la página, añadirá mensaje de error y se podrá apostar de nuevo con las monedas que tienes disponibles
Resultado obtenido	Se recarga la página, se añade mensaje de error y se puede apostar de nuevo con las monedas que tienes disponibles

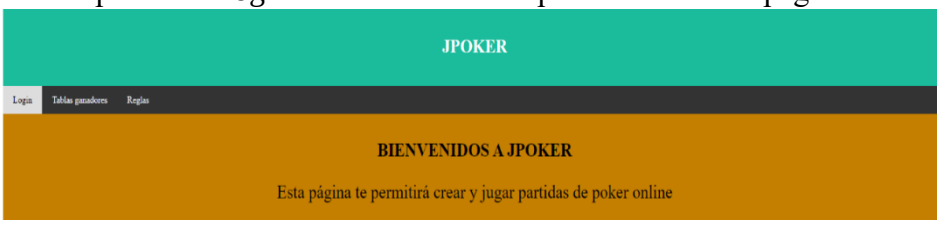
PCB-08	**Entrar en partida**
Propósito	No poder entrar en partida si no tienes monedas suficientes para hacer la apuesta inicial
Prerrequisitos	Buscar, crear o unirse a una partida, ya sea pública, privada o contra el ordenador y hacer la apuesta inicial.
Datos de entrada	
Resultado esperado	Se redirigirá a la página a la página de inicio, añadirá mensaje de error
Resultado obtenido	Redirección a la página de inicio con el mensaje de error

9 Manuales

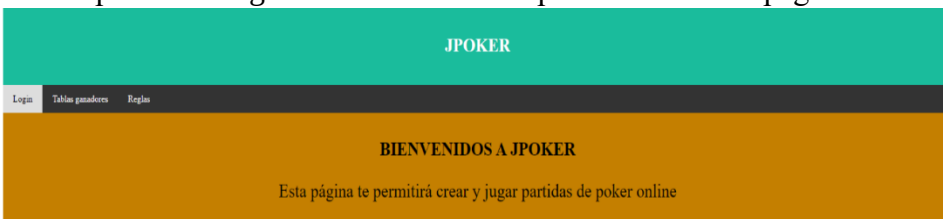
9.1 Manual de usuario

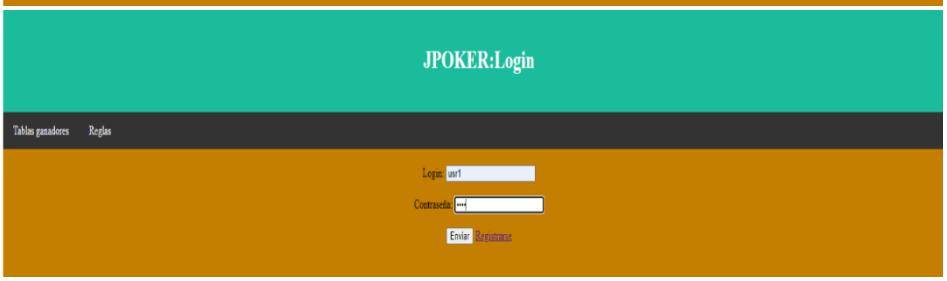
Para entrar al portar web tendremos que entrar en http://127.0.0.1:8000/poker/

- Login en la aplicación
  1. Dar opción "Login" del menú superior de la página web.

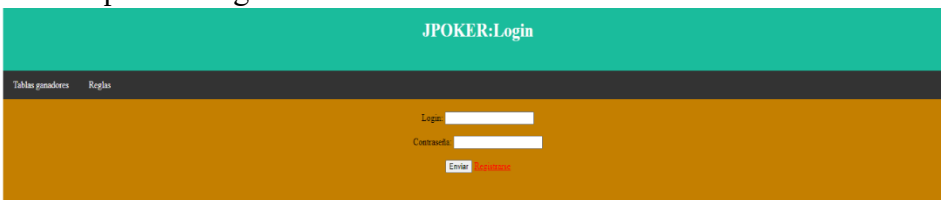


  2. Introducir login y contraseña.
  3. Dar botón de "Enviar".
- Registro en la aplicación
  1. Dar opción "Login" del menú superior de la página web.

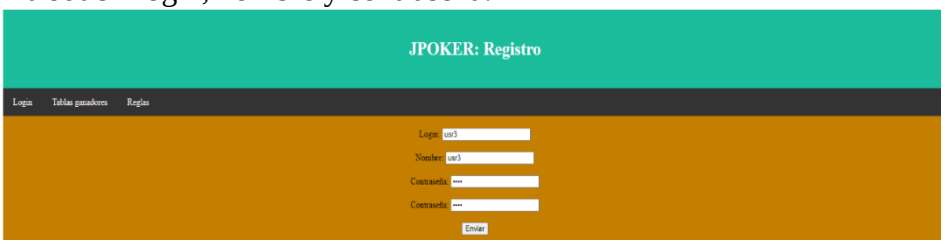




  2. Pulsar opción "Registrarse".

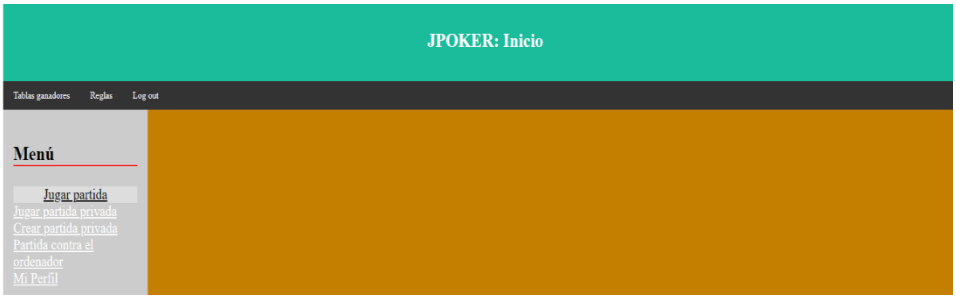


  3. Introducir login, nombre y contraseña.

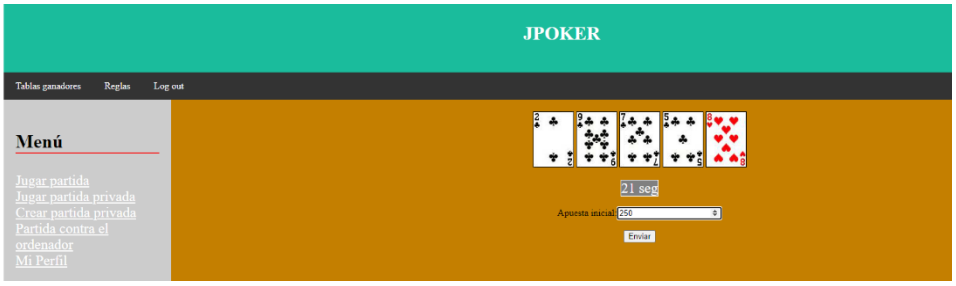


  4. Dar botón de "Enviar".

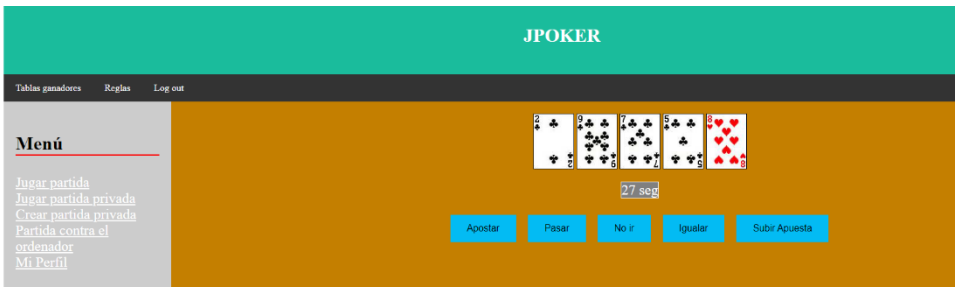
- Jugar partida
    1. Dar opción “Jugar partida” del menú a la izquierda de la página web.



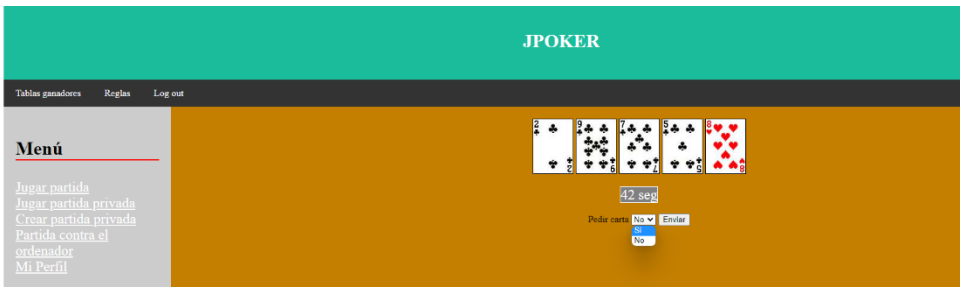
    2. Ver mano y realizar apuesta inicial.



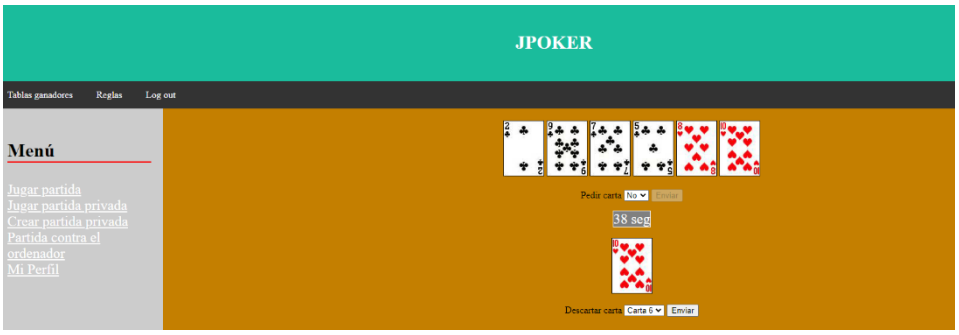
    3. Se muestra menú de la segunda ronda y se escoge alguna de las opciones de la ronda.



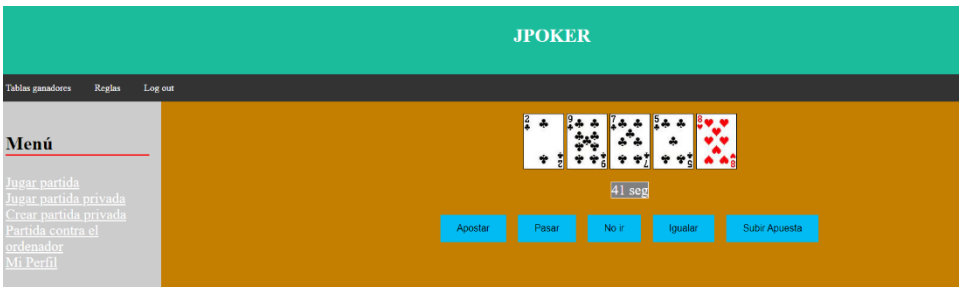
    4. Se muestra un menú que te permite pedir o no carta.



5. Si el jugador pide carta tendrá la opción de descartar una de las cartas de la mano.

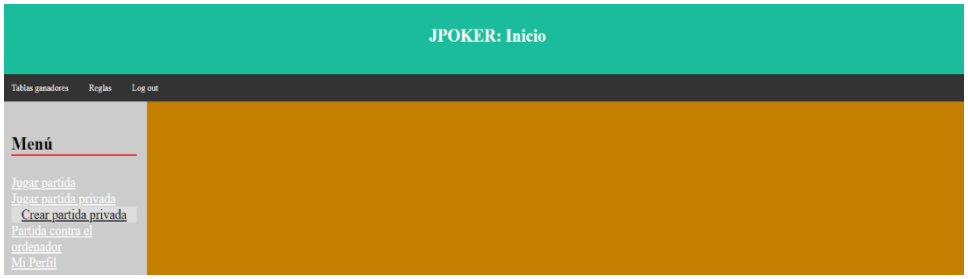


6. Se muestra menú de la segunda ronda y se escoge alguna de las opciones de la ronda.



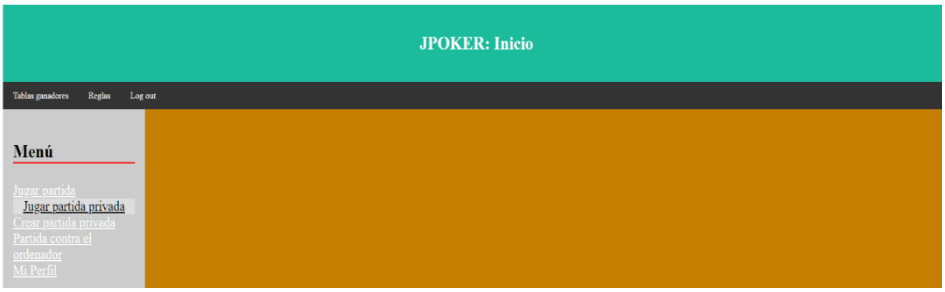
7. Ver ganador de la partida

- Crear partida privada
    1. Dar opción “Crear partida privada” del menú a la izquierda de la página web.



    2. Añadir el número de jugadores.

    3. Compartir código de partida.
    4. Pulsar botón “Entrar partida”.
- Unirse partida privada
    1. Dar opción “Jugar partida privada” del menú a la izquierda de la página web.

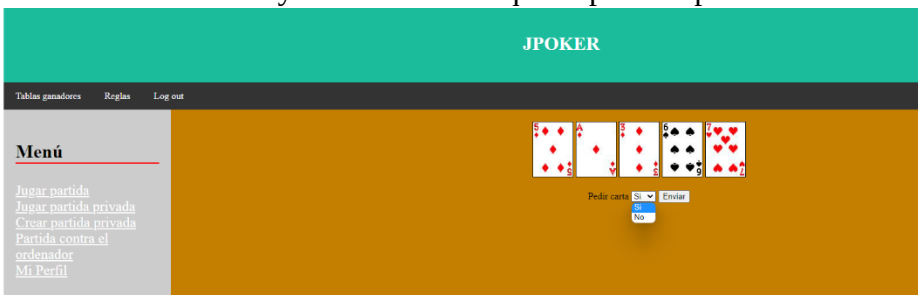


    2. Introducir código partida.

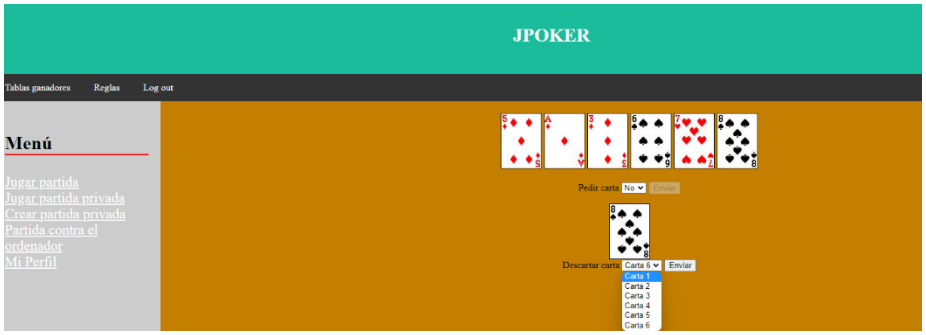
    3. Pulsar botón “Enviar”.

- Jugar partida contra el ordenador
  1. Dar opción “Partida contra el ordenador” del menú a la izquierda de la página web.

  2. Se muestra tus cartas y tienes un menú que te permite pedir o no carta.



  3. Si el jugador pide carta tendrá la opción de descartar una de las cartas de la mano.

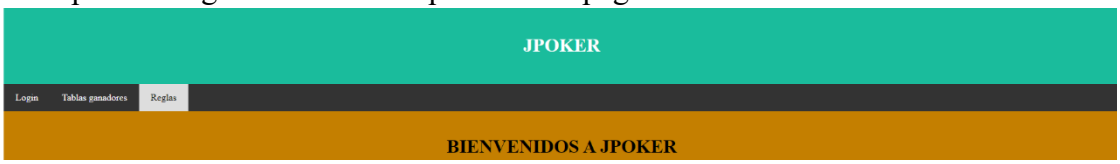


  4. Ver ganador de la partida

- Ver mi perfil
    1. Dar opción “Mi perfil” del menú a la izquierda de la página web.

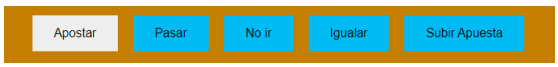
- Ver tabla con top 10 de ganadores
    1. Dar opción “Tabla ganadores” del menú superior de la página web.

- Ver reglas
    1. Dar opción “Reglas” del menú superior de la página web.



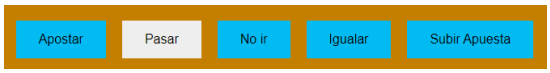
- Apuesta iniciar
    1. Introducir apuesta inicial, mayor que 10, que es el mínimo.

    2. Pulsar enviar.
- Apostar
    1. Pulsar botón “Apostar”.



    2. Introducir apuesta.

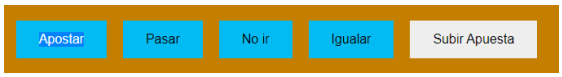
    3. Pulsar botón “Enviar”.
- Pasar
    1. Pulsar botón “Pasar”.



- No ir
    1. Pulsar botón “No ir”

- Igualar
    1. Pulsar botón “Igualar”

- Subir apuesta
    1. Pulsar botón “Subir apuesta”.



    2. Introducir apuesta.

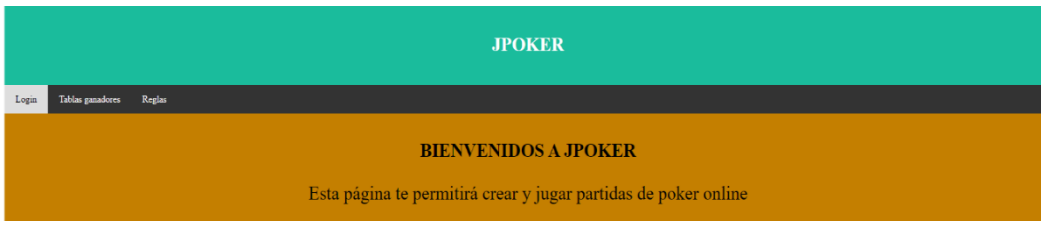
    3. Pulsar botón “Enviar”.

- Pedir carta
    1. Escoger opción “Si” o “No”.
    2. Pulsar botón “Enviar”.

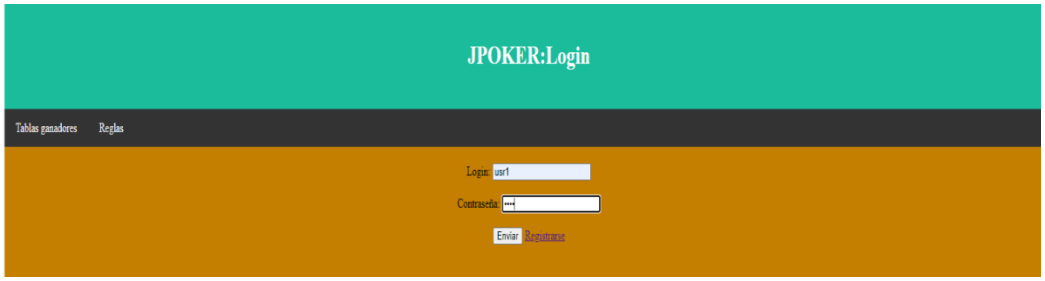
- Descartar carta
    1. Escoger una de las 6 cartas de la mano tras pedir una carta
    2. Pulsar botón “Enviar”.

9.2 Manual de administrador

- Login en la aplicación
  1. Dar opción “Login” del menú superior de la página web.

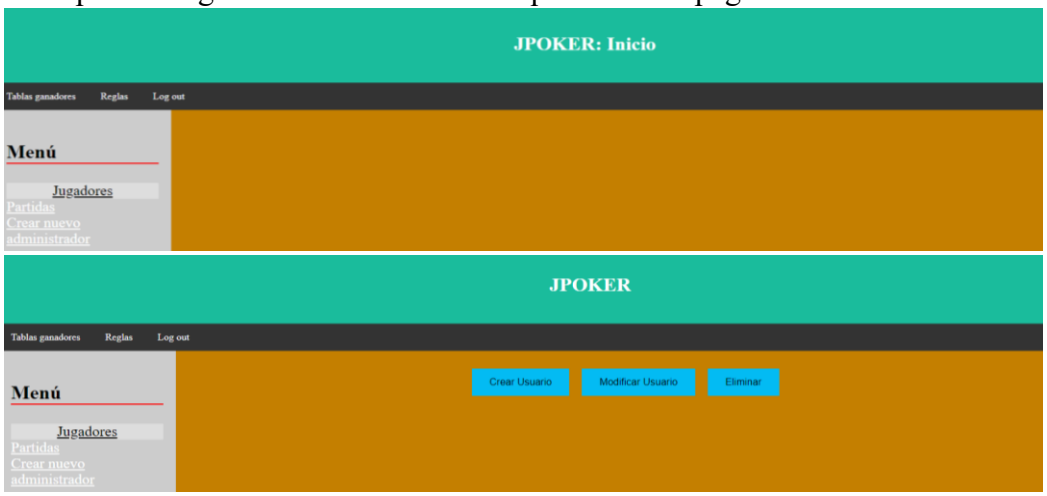


  2. Introducir login y contraseña.



  3. Dar botón de “Enviar”.

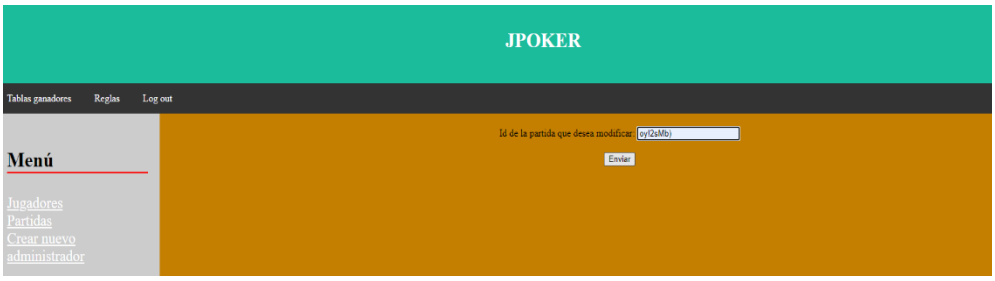
- Administrar Jugador
  1. Dar opción “Jugadores” del menú a la izquierda de la página web.



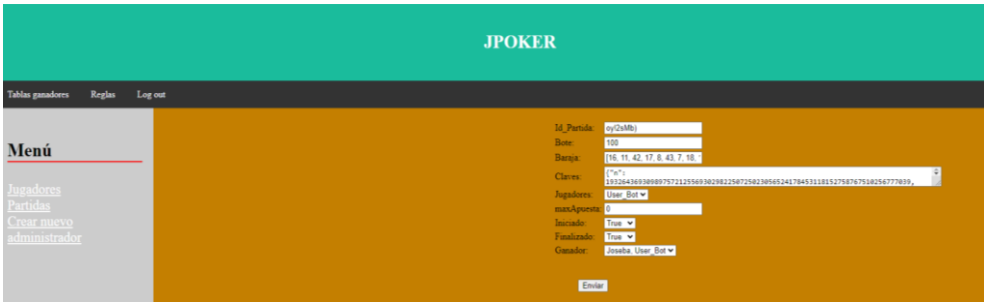
  2. Elegir cualquiera de las 3 opciones disponibles

- Administrar partida.
  1. Dar opción “Partidas” del menú a la izquierda de la página web.

2. Introducir código partida



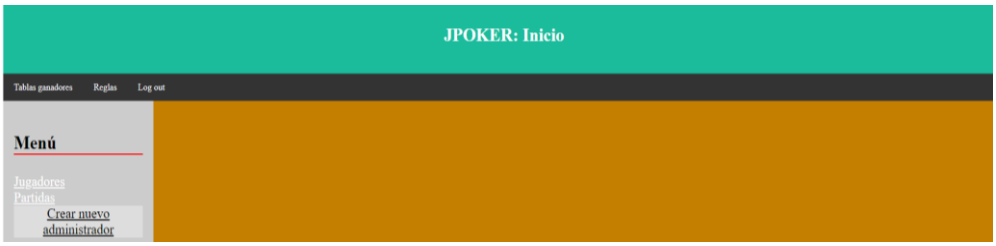
3. Modificar datos de la partida



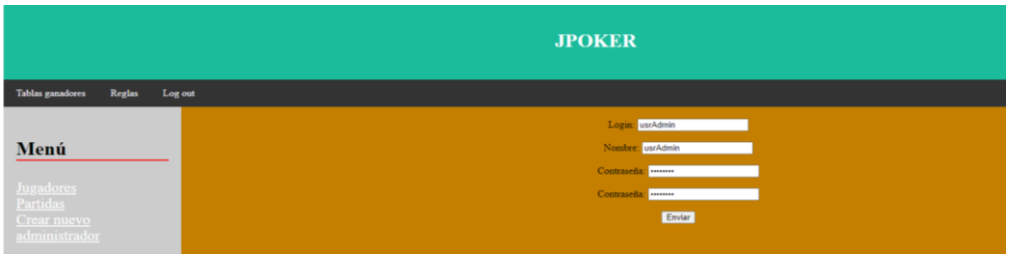
4. Dar botón de “Enviar”.

➢ Registrar nuevo administrador

1. Dar opción “Crear nuevo administrador” del menú a la izquierda de la página web.



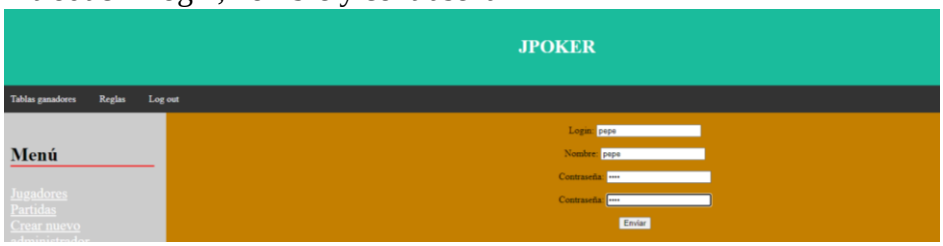
2. Introducir login, nombre y contraseña.



3. Dar botón de “Enviar”.

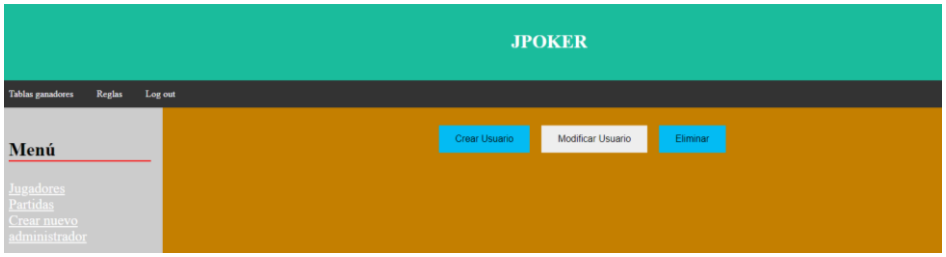
- Crear nuevo jugador
  1. Tras pulsar el botón “Jugadores” del menú lateral, pulsar la opción “Crear Usuario”.

  2. Introducir Login, nombre y contraseña

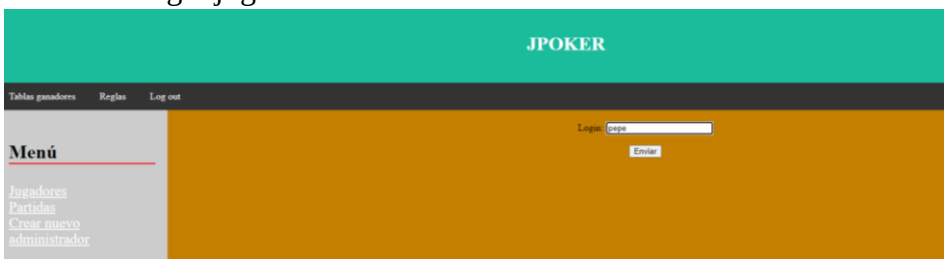


  3. Dar botón de “Enviar”.

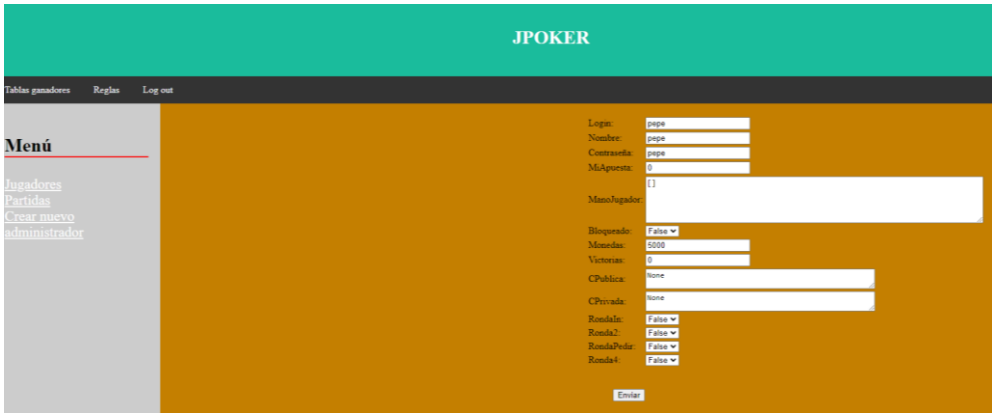
- Modificar jugador
  1. Tras pulsar el botón “Jugadores” del menú lateral, pulsar la opción “Modificar Usuario”.



  2. Introducir login jugador



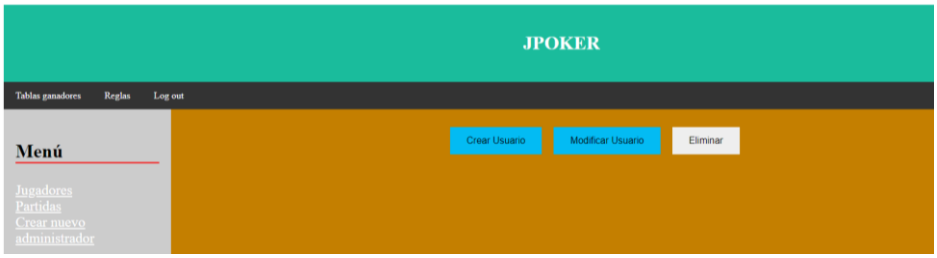
3. Modificar datos del jugador



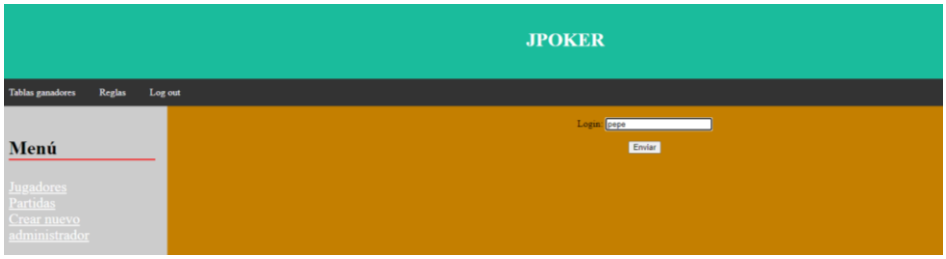
4. Dar botón de “Enviar”.

➢ Eliminar jugador

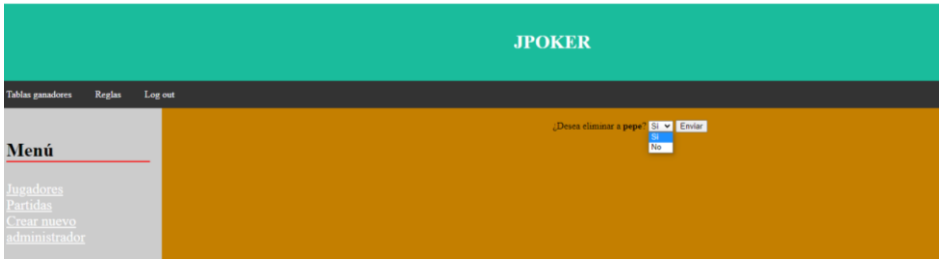
1. Tras pulsar el botón “Jugadores” del menú lateral, pulsar la opción “Eliminar”.



2. Introducir login jugador



3. Confirmar eliminación del jugador



4. Dar botón de “Enviar”.

Por otra parte, la aplicación desarrollada utiliza el sistema de autenticación de que provee Django que además provee de un sistema de administración al que se puede acceder mediante el siguiente enlace http://127.0.0.1:8000/admin/. Hay creado un usuario con las siguientes credenciales:

- **Login**: Admin1
- **Email**: Admin1@admin.com
- **password**: Admin1

9.3 Manual de instalación

1. Instalación del programa “Visual Studio Code”
   En el siguiente enlace puede obtenerse el programa
   https://code.visualstudio.com/download

2. Instalar MYSQL
   Será necesario añadir lo siguiente:
   - **Usuario**: root
   - **Contraseña**: 1234Casa
   - **Puerto**: 3306

   En el siguiente enlace puede obtenerse el programa
   https://dev.mysql.com/downloads/installer/
   Una vez instalado, se deberá abrir y ejecutar el fichero que hay en la carpeta sql mencionada en el apéndice.

3. Instalar la extensión de Python

4. Instalar Python
   En el siguiente enlace puede obtenerse el programa:
   https://www.python.org/downloads/
   Se deberá instalar la versión 3.9

5. Creamos el entorno virtual
   5.1 Instalamos el entorno virtual
   python3 -m venv .venv
   5.2 Abrir la carpeta del proyecto
   5.3 Seleccionamos el intérprete “.\env\Script\python.exe”
   5.4 Ejecutar la terminal
   5.5 Ejecutamos el comando *.venv/script/activate*
   5.6 Actualización de PIP
   python -m pip install –upgrade pip
   5.7 Instalamos Django en el entorno virtual con el siguiente comando:
   python -m pip install django

6. Instalar la extensión de MYSQL

7. Nos movemos en la terminal a la carpeta del proyecto

8. Instalamos módulo PYMYSQL

   Ejecutamos pip3 install PyMySQL en la terminal.

9. Instalamos módulo Sympy

   Ejecutamos *pip install sympy* en la terminal.

10. Instalamos módulo *psycopg2*

    Ejecutamos *pip3 install psycopg2* en la terminal.

11. Instalamos módulo JSON

    Ejecutamos *pip install jsonfield* en la terminal.

12. Instalamos módulo preventconcurrentlogins

    Ejecutamos *pip install django-preventconcurrentlogins*.

13. Creamos un superusuario de Django: Para ello escribimos en la terminal lo siguiente: *python manage.py createsuperuser*

    Y las credenciales que añadamos son las siguientes:

- **Username**: Admin1
- **Email address**: Admin1@admin.com
- **Password**: Admin1

10 Conclusiones

10.1 Conclusiones generales

A lo largo de este proyecto se han puesto en práctica los fundamentos básicos y los conocimientos aprendidos acerca los lenguajes como HTML, CSS, JavaScripts y MYSQL, pero principalmente PYTHON.

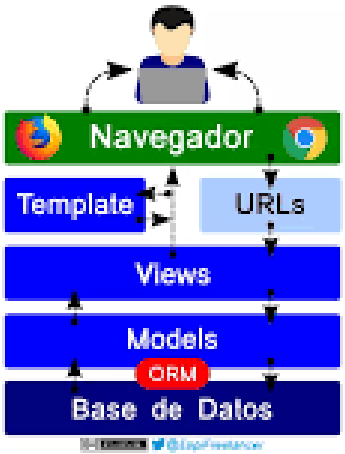
Se han confeccionado diferentes diagramas arquitectónicos, basándonos concretamente en el estilo arquitectónico de aplicaciones web por capas, que es el que mejor se ajusta a nuestro caso práctico.

Se ha utilizado el patrón MTV (Model Templete View).
En Django, el controlador está presente, nada más que de una manera intrínseca, ya que todo el framework Django es el controlador.

- **Model**: Maneja todo lo relacionado con la información, esto incluye como acceder a esta, la validación, relación entre los datos y su comportamiento.

- **View**: Es un enlace entre el modelo y el template. Decide qué información será mostrada y por cual template.

- **Template**: Decide como será mostrada la información.



Cabe lugar hacer hincapié en la importancia que tiene el uso de la criptografía por motivos de seguridad, ya que, aunque no sea nuestro caso, se suelen manejar altas cantidades de dinero. Pero no solo es importante por eso, la criptografía hace que el juego sea más justo y que todo el mundo juegue en igualdad de oportunidades ya que conocer las cartas del resto de jugadores daría una enorme ventaja respecto del resto.

La criptografía ha sido aplicada en las partidas tanto en el reparto inicial de las cartas como en la ronda de la partida en la que se puede solicitar carta. En ambas ocasiones, inicialmente la baraja es cifrada, con su clave pública, por uno de los jugadores, el otro jugador, al que se quiere repartir las cartas, recibe la baraja y la cifra, también con su clave pública. Tras cifrar la baraja, escoge cinco cartas (o una si es la ronda en la que se pide carta) le devuelve tanto la baraja como las cartas escogidas al jugador que la cifró. Este jugador las descifra, con su clave privada, y como están cifradas por el otro jugador, este no conoce las cartas que ha escogido, algo importante para no tener cierta ventaja. El jugador recibe la baraja y sus cartas (su mano durante la partida) y las almacena. Y como ya se ha explicado en el apartado, en caso de haber más jugadores, este jugador haría lo mismo con el siguiente jugador y así sucesivamente.

En nuestra plataforma web solo es utilizada durante la partida. Sin embargo, sería interesante, utilizando otros algoritmos, el cifrado de los datos almacenados que, aunque en nuestro caso no son comprometedores, si se conocen, podrían llegar a incurrir en distintos delitos.

10.2 Líneas de trabajo futuras

Como servicio web recién desarrollado, el proyecto puede mejorar, entre estas mejoras están:

- Llamadas a servicios web externos.
- Mejora de las interfaces del servicio web.
- Cambiar de las monedas utilizadas en el juego por dinero real, esto implicaría poder realizar consultas a bases de datos externas.
- Compartir resultados en las redes sociales.
- Chat de mensajes entre los jugadores de la partida, esto implicaría la implementación de sockets.
- Mejora del servidor web.
- Aumento del uso de la criptografía para mayor seguridad del usuario.

Nota: Actualmente, la aplicación solo se puede utilizar desde una única terminal, para poder jugar desde distintas terminales habría que configurar un servidor. Sin embargo, una cosa indispensable para ello es tener una IP Fija para el servidor, recurso hardware indispensable para poder probarlo entre terminales con distintas conexiones.

11 Bibliografía

11.1 Bibliografía

J. I. Farrán: Computación Matemática con Python, Universidad de Valladolid (2020).
A. Fúster et al.: Técnicas criptográficas de protección de datos (3ª edición), Ed. Ra-Ma (2004).
M. Lutz: Learning Python, O'Reilly (2014).
S. McNeely: Ultimate Book of Card Games, Chronicle Books (2009)
D. Rubio: Beginning Django, Apress (2017).
B. Schneier: Applied Cryptography, John Wiley & Sons (1996).
M. Tsitoara: Beginning Git and GitHub, Apress (2019).

T. Connolly, C, Begg. Database Systems: A Practical Approach to Design, Implementation, and Management. Addison-Wesley, 2015. 6a Ed.

R. Elmasri, S.B. Navathe. Fundamentals of Database Systems. Addison-Wesley, 2011. 6a Ed.

A. Silberschatz, H.F. Korth, S. Sudarshan. Database System Concepts. McGraw-Hill, 2006. 5a Ed.

Shklar, L. and Rosen, R. Web application Architecture, 2nd ed. Wiley.

Uniform Resource Identifier (URI): Generic Syntax [RFC3986]

K. Wiegers, J. Beatty. Software Requirements. Microsoft, 2013. 3aEdición

I. Sommerville. Software Engineering. Addison-Wesley, 2015. 10aEdición

P.Bourque, R.E. Fairley. Guide to the Software Engineering Body of Knowledge. IEEE Computer Society, 2014. Versión 3.0.

J. Beatty, A. Chen. Visual Models for Software Requirements. Microsoft, 2012. 1a Edición

G. Kotonya, I. Sommerville. Requirements Engineering: Processes and Techniques. John Wiley and Sons, 1998. 1a Edición

11.2 Webgrafía

https://www.pokerstars.es/poker/games/rules/

https://www.pokerstars.es/poker/

https://okdiario.com/howto/como-jugar-poker-2448058

https://gist.github.com/JJ/dac9c3afa29019b5a8764dc6c1c00efc

https://github.com/steveshambles/Hi-Lo-card-game

https://docs.djangoproject.com/en/3.1/topics/install/#installing-distribution-package

https://www.youtube.com/watch?v=2DbWqyBR4Oo

https://code.visualstudio.com/docs/python/tutorial-django

https://repositorio.unican.es/xmlui/bitstream/handle/10902/3107/Javier%20Gonzalez%20Villa.pdf?sequence=1&isAllowed=y

https://riunet.upv.es/bitstream/handle/10251/74152/TORREGROSA%20-%20Desarrollo%20de%20un%20juego%20multijugador%20y%20multiplataforma.pdf

https://elbauldelprogramador.com/introduccion-django-instalacion-y-primer-proyecto/

https://www.javaer101.com/es/article/897811.html

https://www.desarrollolibre.net/blog/django/configurar-y-conectar-una-base-de-datos-en-un-proyecto-en-django#.YHRWgOgzaUk

https://www.srcodigofuente.es/tutoriales/ver-tutorial/como-almacenar-recuperar-array-base-de-datos

https://es.stackoverflow.com/questions/100447/c%C3%B3mo-puedo-cargar-archivos-css-y-js-en-django

https://docs.hektorprofe.net/django/web-personal/template-tag-url/

https://djangotutorial.readthedocs.io/es/1.8/intro/index.html

https://www.youtube.com/watch?v=eTrFc27lWrU

https://codigofacilito.com/articulos/mejora-consultas-django

https://www.youtube.com/watch?v=DAGS_EANRac

https://stackoverflow.com/questions/43180977/python-django-mysql-issue-with-updating-form-table-in-html-page-without-using

https://uniwebsidad.com/libros/django-1-0/capitulo-5/tu-primera-aplicacion

https://elbauldelprogramador.com/crear-formularios-en-django-partir-de-un-modelo-con-modelform/

https://developer.mozilla.org/es/docs/Learn/Server-side/Django/Authentication

http://django-book.blogspot.com/2012/11/realizando-consultas-una-vez-que-hayas.html

https://developer.mozilla.org/es/docs/Learn/Server-side/Django/Forms

https://developer.mozilla.org/es/docs/Learn/Server-side/Django/Sessions

https://developer.mozilla.org/es/docs/Learn/Server-side/Django/Home_page

https://uniwebsidad.com/libros/django-1-0/capitulo-12/el-entorno-de-sesiones-de-django

https://realpython.com/python-sockets/

https://github.com/techwithtim/Network-Game-Tutorial

https://pythontic.com/modules/socket/udp-client-server-example

https://developer.mozilla.org/es/docs/Learn/Server-side/Django/Models

http://es.uwenku.com/question/p-haqtqdeg-z.html

https://tutorial.djangogirls.org/es/django_models/

https://cursos.mejorcodigo.net/article/como-guardar-objetos-json-en-mysql-30

https://norfipc.com/web/javascript-facil-paginas-web-ejemplos.html

https://www.arkaitzgarro.com/javascript/capitulo-16.html

https://francescricart.com/formulario-que-se-envia-de-forma-automatica-al-rellenar-los-datos/

https://developer.mozilla.org/es/docs/Web/API/EventTarget/addEventListener

https://docs.djangoproject.com/en/3.2/ref/contrib/postgres/fields/

https://www.youtube.com/watch?v=LbdUpY1I1zg

http://es.uwenku.com/question/p-hyqnzjte-bn.html

https://stackoverflow.com/questions/49099565/accessing-a-list-contained-in-a-json-object-with-python/49228857

https://docs.djangoproject.com/en/1.8/topics/auth/default/#authentication-data-in-templates

https://stackoverflow.com/questions/34286973/prevent-multiple-logins-using-the-same-credentials

https://medium.com/@a01207543/django-conecta-tu-proyecto-con-la-base-de-datos-mysql-2d329c73192a

https://ed.team/blog/como-trabajar-con-json-en-mysql

Apéndice 1

Algoritmo de cifrado

El algoritmo utilizado para cifrar las cartas durante el transcurso de la partida es el RSA (Rivest, Shamir y Adleman), que es un algoritmo asimétrico.

El código que he implementado para realizar el cifrado en el servicio web es el siguiente:

Antes de todo, se han importado las librerías necesarias:

```
from copy import deepcopy
import sympy
from sympy import randprime, isprime
from random import randint
```

Tras ello, lo primero hay que generar las claves, para ello se ha realizado lo siguiente:

Como ya explicamos en el punto 7 del documento, necesitamos 2 números primos seguros, se ha implementado el siguiente método:

```
def generarPrimoSeguro(e):
    p = randprime(2**(e-25), 2**(e+25))
    while(not isprime(2*p+1)):
        p = randprime(2**(e-25), 2**(e+25))
    return p
```

Para asegurar que sea más seguro, se escogen aleatoriamente dentro de un rango de valores asegurándonos siempre que sea primo. Con estos números primos generamos n y phi, importantes a la hora de generar las claves.

```
def generarN():
    p, q = generarPrimoSeguro(100), generarPrimoSeguro(100)
    n = p * q
    phi = phieuler(p, q)
    return {'n': n, 'phi': phi}
```

```
def phieuler(p,q):
    return (p-1) * (q-1)
```

Tras ello calculamos la clave pública (e), que está entre 1 y pide tal forma que el MCD(e,phi) = 1:

```
def obtenere(n):
    t = randint(1,n)
    while (egcd(t,n)[0] != 1):
        t = randint(1,n)
    return t
```

```
def egcd(a, b):
    x, X = 0, 1
    y, Y = 1, 0
    while (b != 0):
        B = b
        q = a // b
        a, b = b, a % b
        x, X = X - q * x, x
        y, Y = Y - q * y, y
    return ([B, X, Y])
```

Tras ello calculamos la clave privada (d) a través del inverso modular:

$$di \equiv e{-}1\ i\ (\text{mod}\ \Phi(ni))$$

Asegurándonos que se cumpla:

$$ei \cdot di \bmod phi = 1$$

```
def modInverse(a,m):
    L = egcd(a,m)
    return L[1] % m
```

Por último, tras el proceso de generación de claves, el algoritmo de cifrado:

```
def mpow(x, y, z):
    x = int(x)
    y = int(y)
    e = 1
    while y:
        if y & 1:
            e = e * x % z
        y >>= 1
        x = x * x % z
    return e
```

Donde "**x**" es lo que queremos cifrar, "**y**" es la clave (pública o privada, dependiendo de si queremos cifrar o descifrar) y "z" es el valor de n calculado en el proceso de generación de claves. El valor de retorno "**e**" es el valor de "**x**" ya cifrado (o descifrado).

Apéndice 2

Contenido del repositorio

El repositorio en el que está publicado el trabajo consta de los siguientes documentos:

- **Memoria**: La memoria contiene una explicación detallada del proyecto realizado
- **Proyecto TFG**: Esta carpeta contiene el programa desarrollado para el proyecto y que consta de los siguientes documentos:

  - **.vscode**
  - **Jpoker**
    - **_init_:** un archivo vacío que le dice a Python que esta carpeta es un paquete de Python.
    - **asgi:** un punto de entrada para servidores web compatibles con ASGI para servir su proyecto. Por lo general, deja este archivo como está, ya que proporciona los enlaces para los servidores web de producción.
    - **settings:** contiene la configuración del proyecto Django, que modifica durante el desarrollo de una aplicación web.
    - **urls:** contiene una tabla de contenido para el proyecto Django, que también modificas en el curso del desarrollo.
    - **wsgi:** un punto de entrada para servidores web compatibles con WSGI para servir su proyecto. Por lo general, deja este archivo como está, ya que proporciona los enlaces para los servidores web de producción.
  - **Lib**
  - **Poker**
    - **migrations:** carpeta utilizada por la utilidad administrativa de Django para administrar las versiones de la base de datos.
    - **vistas:**
      - **Cartas:** Imágenes de las cartas utilizadas en la aplicación.
      - **css:** Código .css utilizado en la aplicación.
      - **html:** Código .html utilizado en la aplicación.
      - **js:** Código .js utilizado en la aplicación.
      - **python:** Código .python utilizado en la aplicación.
      - **sql:** Código .sql utilizado en la aplicación.
      - **index:** Página html inicial.
    - **_init_**
    - **admin:** archivo para crear una interfaz administrativa.
    - **apps:** archivo que contiene la configuración de la aplicación
    - **models:** archivo que contiene clases que definen sus objetos de datos
    - **tests:** archivo para crear pruebas
    - **urls:** archivo en donde especifica patrones para enrutar diferentes URL a sus vistas.

- **views:** archivo que contiene las funciones que definen páginas en su aplicación web

❖ **Scripts**
❖ **db**: base de datos SQLite predeterminada
❖ **manage.py**: La utilidad administrativa de línea de comandos de Django para el proyecto. Ejecuta comandos administrativos.
❖ **pyvenv.cfg**