Manual de Corrección

---

1. Introducción

Este documento tiene como objetivo proporcionar una guía clara y concisa para revisar y validar la funcionalidad de la aplicación **007 Casino**. A lo largo del manual, se describen las principales características del proyecto, incluidas las credenciales de acceso de prueba, las áreas clave donde se han implementado conceptos avanzados como hilos y recursividad, y otras funcionalidades destacadas que requieren evaluación durante el proceso de corrección.

IAG: Este documento ha sido generado con ayuda de la herramienta de inteligencia artificial ChatGPT con tecnología GPT-4o.

---

2. Credenciales de Acceso

La aplicación permite a los usuarios acceder mediante la creación de una cuenta nueva o utilizando las credenciales de ejemplo proporcionadas a continuación.

- **Crear un nuevo usuario:**
  - Presione el botón **"Registrar"** en la pantalla de inicio de sesion.
  - Complete los campos obligatorios como nombre, apellidos, DNI...
  - Una vez que se validen los datos, el usuario podrá acceder al sistema.
- **Credenciales de prueba:**
  Para realizar pruebas rápidas de la funcionalidad del sistema, utilice las siguientes credenciales preconfiguradas:
  - **Usuario:** `test`
  - **Contraseña:** `te`

---

3. Uso de Hilos (Threads)

El uso de hilos en el proyecto permite gestionar tareas que demandan un uso intensivo de recursos o que requieren animaciones fluidas sin bloquear la interfaz gráfica (UI). A continuación, se detalla su aplicación en los distintos módulos:

a. Blackjack - Cálculo de Probabilidades

Ubicación: `gui.juegos.blackjack.PanelBlackjack.<init>`

- **Descripción:**
  El cálculo recursivo de probabilidades en el Blackjack es una operación intensiva en términos de computación. Para evitar que esta tarea afecte la fluidez de la interfaz gráfica, se ejecuta en

un hilo separado. Esto asegura que los usuarios puedan interactuar con la aplicación sin interrupciones mientras se realizan los cálculos.

b. Blackjack - Dealer muestra cartas

Ubicación: `gui.juegos.blackjack.LogicaBlackjack.plantarseJugador`

- **Descripción:**
  Cuando el jugador decide plantarse, el Dealer comienza a robar cartas. Este proceso se anima utilizando un hilo que introduce un intervalo de 1 segundo entre cada carta, simulando un comportamiento realista del juego y mejorando la experiencia del usuario.

c. Carreras de Caballos - Movimiento de Caballos

Ubicación: `gui.juegos.caballos.PanelCaballos.Hilo.run`

- **Descripción:**
  Al iniciar una carrera, cada caballo avanza de manera independiente en la pista. Un hilo controla el movimiento estos caballos, generando un avance aleatorio cada 50 milisegundos. Este enfoque permite una animación fluida y dinámica durante la carrera.

d. Dino Run - Animación del Dino

Ubicación: `gui.juegos.dinoRun.PanelDino.startAnimation`

- **Descripción:**
  La animación del Dino corriendo se gestiona a través de un hilo que actualiza los cuadros de la animación. Esto garantiza un movimiento continuo y fluido mientras se evita que la actualización gráfica bloquee la interacción del usuario.

e. Ruleta - Girar la Ruleta

Ubicación: `gui.juegos.ruleta.PanelRuleta.spinRoulette`

- **Descripción:**
  Cuando el jugador pulsa el botón para girar la ruleta, se inicia un hilo que controla la animación del giro. Este hilo asegura que el movimiento sea progresivo y visualmente atractivo, proporcionando una experiencia de usuario más realista.

f. Slots - Animación de los Rodillos

Ubicación: `gui.juegos.slots.LogicaSlots.Hilo.run`

- **Descripción:**
  Al jugar una partida de slots, cada una de las tres columnas comienza a girar de manera independiente. Un hilo controla el giro estas columnas, deteniéndolas secuencialmente: primero la primera columna, luego la segunda y finalmente la tercera. Este enfoque mejora significativamente la calidad de la animación en comparación con un giro simultáneo de las tres columnas.

---

4. Uso de Recursividad

El proyecto emplea recursividad para resolver problemas que involucran cálculos complejos y tareas repetitivas, maximizando la claridad del código en ciertas áreas clave. A continuación, se describen los casos de uso:

a. Blackjack - Cálculo de Probabilidades

Ubicación: `gui.juegos.blackjack.PanelBlackjack.calcularProbabilidades`

- **Descripción:**
  Se utiliza un algoritmo recursivo para calcular todas las combinaciones posibles que pueden surgir durante una partida de Blackjack. El objetivo es determinar las probabilidades de ganar, empatar o perder si el jugador decide pedir otra carta o plantarse.
- **Cómo Probarlo:**
  - Inicia una partida de Blackjack.
  - Haz clic en el botón **"Calcular"** ubicado en la esquina superior derecha.
  - Ten paciencia, ya que el cálculo puede tardar un tiempo significativo, especialmente al inicio de la partida, debido a la cantidad de combinaciones que deben analizarse.
  - **Sugerencia:** Realiza el cálculo cuando el jugador tenga un puntaje alto (18 o más). Esto reduce el número de combinaciones posibles y acelera el proceso.

b. Dino Run - Cálculo del Multiplicador

Ubicación: `gui.juegos.dinoRun.PanelDino.calculateMultiplier`

- **Descripción:**
  Esta función recursiva realiza un cálculo simple para determinar el multiplicador de premio basado en el desempeño del jugador. El resultado se aplica como un porcentaje adicional sobre lo apostado, recompensando la habilidad del jugador.

5. Otras Funcionalidades

El proyecto **007 Casino** incluye una amplia gama de características que enriquecen la experiencia del usuario y facilitan la gestión del sistema. A continuación, se detallan:

---

a. Información de Datos Incorrectos

- **Funcionalidad:**
  Al registrar o modificar datos de un usuario, se valida la información ingresada para evitar errores. Si los datos son incorrectos, se desactiva el botón "Aceptar" y se muestra un mensaje indicando el problema.
- **Ejemplos de Validaciones:**
  - **DNI:** Debe tener 8 dígitos seguidos de una letra.

    - **Fecha de Nacimiento:** Verifica que el usuario sea mayor de edad.
- **Aplicación en Transacciones:**
  Se aplican las mismas validaciones al depositar o retirar dinero, asegurando que los valores sean correctos.

---

b. JTable de Historial de Movimientos

- **Funcionalidad:**
  En el menú del perfil, los usuarios pueden inspeccionar un historial detallado de sus movimientos.
- **Características:**
    - **Interactividad:**
        - Al pasar el ratón por una línea, su color se aclara.
        - Al seleccionar una línea, se abre un panel con información detallada.
    - **Columnas Personalizadas:**
        - "Tipo" y "Acción" incluyen imágenes que representan el juego y el resultado.
        - "Modificación" cambia de color según el impacto: verde (positivo), rojo (negativo), amarillo (neutro).
        - El texto cambia de color según el tipo de movimiento:
            - Magenta: Premio de bienvenida.
            - Azul: Depósito.
            - Amarillo: Retiro.
            - Rojo: Apuesta.
    - **Hora:**
        - El fondo varía entre oscuro y claro según la hora del movimiento. El color del texto se adapta para garantizar la legibilidad.
    - **Ordenamiento:**
        - Los datos pueden ordenarse pulsando las cabeceras de las columnas.

---

c. Atajos de Teclado

- **Funcionalidad:**
  Los atajos de teclado facilitan la navegación por el menú principal y la salida de juegos.
- **Atajos Disponibles:**
    - **Esc:** Cerrar sesión o salir del juego.
    - **C:** Abrir Carreras de Caballos.
    - **R:** Abrir Ruleta.
    - **S:** Abrir Slots.
    - **B:** Abrir Blackjack.
    - **M:** Abrir Minas.
    - **D:** Abrir Dinosaurio.

---

d. Configuración (`Properties`)

- **Funcionalidad:**
  En el directorio `conf/`, los archivos `.properties` permiten configurar parámetros relacionados con la interfaz y la base de datos.
- **Gestión de Configuración:**
  - Los archivos se gestionan a través de la clase `src/io/ConfProperties.java`.
  - Los cambios realizados durante la ejecución (como activar el modo oscuro) se guardan para mantener la persistencia de las preferencias del usuario.

---

e. Base de Datos

- **Estructura de la Base de Datos:**
  - **`asunto (id, asunto)`:** Lista los tipos de movimientos posibles.
  - **`historial_movimientos (id, fecha, hora, usuario, cantidad, asunto, saldo_final)`:** Registro detallado de los movimientos.
  - **`usuario (usuario, contrasena, fichas, nombre, apellidos, dni, mail, prefijo, telefono, provincia, ciudad, calle, numero_calle, fecha_nacimiento, fecha_registro)`:** Información de los usuarios registrados.

---

f. Uso de Java Collections

- **Funcionalidad:**
  Se utiliza un **Comparator** para ordenar los datos del historial de movimientos en orden cronológico (de más reciente a más antiguo). Esto mejora la organización y claridad de la información mostrada.

---

g. Loggers

- **Funcionalidad:**
  Todas las interacciones con la base de datos y los archivos de configuración `.properties` se registran en archivos de log ubicados en `log/*.log`.
- **Beneficios:**
  - Facilitan la detección y resolución de errores.
  - Proveen un historial detallado de las acciones realizadas, útil para auditorías y depuración.