Proyecto de Aplicación Móvil de Juegos Multiplataforma

GameHub Diversión Inteligente al Alcance de Todos

Autores del Proyecto:

Carlos Uriel Domínguez Peña

Sebastián Zárate Delgado

Jesus Gabriel Abrego Tello

Osvaldo Jahir Zambrano Flores

Versión: 1.0

Fecha: 28 de septiembre de 2025

Área: Desarrollo de Software / Entretenimiento Digital

Contenido

Introducción

A nivel de sistema, planteamos la aplicación como una pila Android clásica, separando con rigor las responsabilidades: capa de presentación declarativa en XML (layouts y estilos), lógica y orquestación en Kotlin (Activities/Adapters/servicios), y persistencia/sesiones con almacenamiento local más un punto de extensión para backend remoto. El criterio fue mantener un acoplamiento bajo entre UI y reglas de negocio, de modo que el diseño pueda iterar sin romper la mecánica de juegos o el flujo de autenticación.

Elegimos XML para el front porque ofrece determinismo de layout, soporte pleno del editor visual, tooling maduro (preview multipantalla, qualifiers, traducciones) y compatibilidad inmediata con Material 3 y componentes de soporte (RecyclerView, AppBarLayout, NavigationView). Kotlin gobierna toda interacción: validaciones, navegación, sincronización de estado, reglas de juego, timers y actualización de vistas.

A nivel de estructura, definimos cuatro áreas claras: autenticación (login/registro), catálogo (repertorio tipo Roblox), módulo de juegos (Blackjack, Pac-Man, Brick Breaker, Clicks123, Ruleta, Snake, Solitario y Sudoku) y una librería compartida (modelos, utilidades, gestión de sesión).

El login es la puerta de entrada. La vista define campos de usuario/contraseña, eye-toggle para password, enlaces de recuperación y dos CTAs (iniciar sesión y crear cuenta). La lógica valida formato, normaliza entrada y, si hay backend, firma TLS y tokeniza sesión; si trabajamos local, emite un “ticket de sesión” persistido. Persistimos la identidad con un gestor de sesión ligero almacenado en preferencias (clave “logged_in” y alias de usuario) porque permite cold start rápido, no bloquea el render de la pantalla y evita dependencias tempranas de red.

Si las credenciales son válidas, la navegación hace forward al catálogo y se cierra el login para no dejarlo en el back stack; si no, se retroalimenta con error semántico (usuario inválido, contraseña corta, etc.). En el registro añadimos validaciones extra (username único, email con MX opcional cuando se conecte servidor) y normalización (lowercase, trim, whitelist de caracteres visibles). El registro guarda un perfil básico (nombre público, sexo opcional, teléfono, correo) y deja el punto de extensión para hash/pepper del password si se conecta a un backend real.

El catálogo reproduce el patrón de Roblox: rail lateral con acciones (Home/Chats/Avatar/Party/More), header con título y avatar, barra de búsqueda, carrusel horizontal de conexiones y grilla responsiva de juegos. Para la lista usamos

RecyclerView con GridLayoutManager (2–3 columnas según ancho), ViewHolders livianos y payloads parciales. Cada tarjeta de juego expone portada, nombre y rating; el rating es porcentual y se redondea a entero para simplificar lectura.

Para previsualización “hover” en móviles reemplazamos el hover por long-press: al presionar prolongado emergen descripción corta y CTAs (“Play”, “Details”) en un overlay semitransparente (FrameLayout) anclado al contenedor y con sombra para separar capas. El scroll principal está en NestedScrollView sólo para la cabecera; las listas en sí no anidan scrolls (los RecyclerView tienen nestedScrollingEnabled=false) para evitar jank. La búsqueda actúa como filtro en memoria (contiene/normaliza) y deja preparado el conector para delegar al servidor cuando exista API de catálogo.

El contador de “Connections” no está hardcodeado. El header muestra el título estático y, a la derecha, un badge reactivo con la cantidad real de amigos. Ese badge se oculta cuando el conteo es cero. El valor se alimenta desde un repositorio de amigos; con backend, lo ideal es exponer un flujo (Flow/LiveData) que emita diffs, y el UI observe para actualizar el badge. Localmente, el repositorio puede ser Room con tabla Friend y DAO que emite count() como Flow; la primera carga lee desde disco y la sincronización de red actualiza la tabla y, por ende, el badge.

Módulos de juegos

- **Blackjack**: definimos un motor de reglas independiente de la UI. El modelo de dominio incluye Shoe (zapato de N mazos), Deck/Shuffle, Hand, Dealer y Player. La evaluación de manos calcula soft/hard con tratamiento correcto de Ases. El dealer actúa con política “hit to 16, stand on all 17s”. El ciclo de ronda es un autómata finito: PlaceBets → InitialDeal → CheckBlackjack → PlayerActions → DealerPlay → Settlement → Payouts → Cleanup.
- **Pac-Man**: implementamos un bucle de juego determinista. La vista de tablero es una custom view que dibuja sobre canvas. El estado del mundo se representa con una grilla matricial. El timestep es fijo (60 Hz). El movimiento de Pac-Man es direccional con buffer de input. Los fantasmas tienen patrones de movimiento y cambian de estado con power pellets.
- **Brick Breaker**: se desarrolló con un canvas interactivo donde la bola rebota sobre ladrillos y paredes. La física de rebote es determinista y las vidas se gestionan por niveles. El puntaje se calcula según los ladrillos destruidos y combos.

- **Clicks123**: juego de reacción rápida. Los botones aparecen de forma aleatoria en la pantalla y el jugador debe tocarlos antes de que desaparezcan. Cada acierto suma puntos y se registra un historial de mejores marcas.
- **Ruleta**: simulamos un casino clásico. El jugador coloca apuestas virtuales en números o colores. La rueda gira con animaciones suaves y el resultado se calcula con RNG. Se gestionan pagos proporcionales según tipo de apuesta.
- **Snake**: tablero cuadriculado donde la serpiente se mueve y crece al comer. Colisiones con paredes o consigo misma terminan la partida. Se registran niveles y puntajes para mejorar la experiencia progresiva.
- **Solitario**: clásico juego de cartas. Maneja baraja, movimiento de cartas entre columnas y reglas estándar de juego. Se controla undo, validaciones de jugadas y detección de victoria.
- **Sudoku**: tablero de 9x9 con celdas para ingresar números. La app valida reglas (fila, columna y subcuadrante), ofrece hints y lleva un seguimiento del tiempo de juego.

A nivel de arquitectura del proyecto, organizamos paquetes por feature: auth, catalog, games.blackjack, games.pacman, games.brickbreaker, games.clicks123, games.ruleta, games.snake, games.solitario, games.sudoku y core (session, models, util, ui). Esto habilita modularización futura en Gradle.

En seguridad, tratamos inputs con normalización y validación estricta; si se habilita backend, las credenciales jamás se guardan en claro. Finalmente, la aplicación combina lo visual (lo que el usuario ve) con la lógica (lo que hace que funcione). Buscamos que cualquier persona pueda usarla sin complicaciones: abrir, iniciar sesión, explorar juegos y jugarlos.

Explicación No Técnica

Cuando empezamos a hacer esta aplicación, lo primero que pensamos fue en cómo queríamos que se viera para el usuario. Imaginamos que al abrirla lo primero sería una **pantalla de inicio de sesión**, muy parecida a las que vemos en cualquier plataforma donde tenemos una cuenta. La idea es que aparezca un cuadro donde escribes tu usuario, otro para la contraseña, y dos botones: uno para entrar y otro para crear una cuenta nueva. Esto asegura que cada persona tenga su propio espacio dentro de la app.

Una vez diseñada esa pantalla, decidimos qué pasa cuando el usuario escribe su información. Si los datos son correctos, la aplicación lo reconoce y le abre la puerta a la siguiente parte: el **repertorio de juegos**. Si los datos no son correctos, aparece un mensaje que avisa que debe intentarlo de nuevo. Además, pensamos en que la aplicación pueda **recordar si ya iniciaste sesión** antes, para no tener que volver a poner usuario y contraseña cada vez que abras la app.

Después de eso, trabajamos en lo que más se parece a Roblox: el **catálogo o repertorio de juegos**. Esta pantalla es básicamente como una tienda de juegos, pero sin compras; es un escaparate. Ahí se muestran las portadas de los juegos, cada una con su nombre y su calificación. La idea es que el usuario pueda ver qué juegos hay disponibles, cuáles son más populares y cuáles podrían llamar su atención. También hay una barra lateral con iconos como **Home, Chats, Avatar, Amigos y Más**, y una barra de búsqueda para encontrar juegos específicos. Además, el apartado de **Connections** muestra tus amigos y se actualiza automáticamente según tus contactos.

Dentro del repertorio, cada juego tiene su propia dinámica:

- **Blackjack**: un juego de cartas estilo casino. El jugador recibe fichas virtuales y puede apostar, pedir cartas o plantarse. El crupier también juega y el resultado se calcula como en un casino real, pero sin dinero real.
- **Pac-Man**: un clásico arcade donde el personaje amarillo se mueve por un tablero comiendo puntitos y evitando fantasmas. Cuando come los puntos grandes, los fantasmas se ponen azules y puede atraparlos.
- **Brick Breaker**: controlas una paleta y debes hacer rebotar la bola para romper todos los ladrillos. A medida que avanzas, los niveles se vuelven más desafiantes y los combos suman puntos.

- **Clicks123**: un juego de reflejos. Botones aparecen de forma aleatoria en la pantalla y hay que tocarlos antes de que desaparezcan. Cada acierto suma puntos y puedes ver tu historial de mejores marcas.
- **Ruleta**: una simulación de ruleta de casino. Puedes apostar en números o colores y ver cómo gira la rueda. El juego calcula las ganancias según el tipo de apuesta.
- **Snake**: manejas una serpiente que crece al comer puntos. Si chocas contra las paredes o contigo mismo, pierdes. Se registran niveles y puntajes.
- **Solitario**: el clásico juego de cartas. Se pueden mover cartas entre columnas siguiendo las reglas del juego y la app valida las jugadas. También puedes deshacer movimientos y ver si ganaste.
- **Sudoku**: tablero de 9x9 donde se ingresan números siguiendo las reglas del Sudoku (fila, columna y subcuadrante). La app ayuda con pistas y lleva un control del tiempo de juego.

Todos los juegos están **conectados al repertorio**, es decir, al tocar un juego en la lista, este se abre de inmediato. Además, si mantienes el dedo sobre un juego por un momento, aparece una vista previa con su descripción y opciones para jugar o ver detalles.

La app completa funciona como un **circuito continuo**: primero el login, luego el catálogo, y finalmente los juegos. Todo está conectado para que la experiencia sea fluida y sencilla. Lo que ves en pantalla está diseñado para ser atractivo, mientras que lo que ocurre "por dentro" se programa para que todo tenga sentido: guardar tu sesión, mostrar juegos reales, ejecutar sus reglas y permitir volver al catálogo siempre que quieras.

En resumen, logramos una aplicación que combina **dos mundos**: lo visual, que es lo que el usuario ve (botones, listas, imágenes), y la lógica, que hace que funcione (validar datos, guardar progresos, calcular puntajes, mostrar resultados). Lo más importante es que cualquier persona, aunque no tenga conocimientos de programación, pueda usarla sin complicaciones: **abrir, iniciar sesión, explorar juegos y jugarlos de forma sencilla y entretenida**.

Marco Teorico

Para desarrollar nuestra aplicación de juegos móviles, nos basamos en **principios fundamentales de ingeniería de software y desarrollo en Android**:

1. **Arquitectura por capas**: Separamos responsabilidades en tres niveles:
    - **Capa de presentación**: Pantallas y elementos visuales (XML), donde definimos layouts, estilos y componentes interactivos.
    - **Capa lógica**: Kotlin, encargado de la gestión de reglas de negocio, navegación entre pantallas, timers y lógica de juegos.
    - **Capa de persistencia**: Almacenamiento local de datos y sesiones, con opción de extender a backend remoto si fuera necesario.
2. **Modularidad y escalabilidad**: Cada juego y función de la app se desarrolló como módulo independiente (Blackjack, Pacman, Brick Breaker, etc.), permitiendo futuras actualizaciones, incorporación de nuevos juegos o conectividad con servicios externos sin afectar el resto de la aplicación.
3. **Experiencia de usuario (UX)**: Aplicamos principios de diseño Material, interfaces intuitivas y consistentes, feedback táctil, y navegación fluida para que la app sea fácil de usar para cualquier persona, incluso sin conocimientos técnicos.
4. **Seguridad y manejo de datos**: Validación estricta de datos de usuario, almacenamiento seguro de información mínima, y preparación para encriptación de credenciales si se implementa backend.
5. **Optimización de rendimiento**: Evitamos sobrecarga de memoria y procesamiento en el renderizado de juegos, asegurando que cada juego funcione de manera fluida, incluso en dispositivos con menor capacidad.

Retos del Proyecto

Durante el desarrollo de los juegos, se presentaron diversas problemáticas que afectaron su avance o funcionamiento. Entre las principales se destacan:

- Problema en la detección de colisiones: en juegos como Brick Breaker, Snake y Pac-Man, fue complejo asegurar que los objetos detectaran correctamente el contacto entre sí
- Optimización de la interfaz en diferentes pantallas: algunos juegos se veían bien en un dispositivo, pero desajustados en otro con diferente resolución
- Control de la lógica del azar: juegos como Ruleta y Blackjack requerían que el azar fuera justo y funcional, evitando repeticiones incoherentes.
- Gestión del flujo del juego: en Solitario y Sudoku, era necesario controlar el paso de fases o validación de jugadas para evitar bloqueos o errores lógicos.
- Sobrecarga de recursos: en juegos con animaciones simples (Pac-Man o Snake), el mal uso de ciclos podía ralentizar la aplicación.

Solución 1: Algoritmos de colisión simplificados

- **Visión del problema:** las colisiones eran inexactas y el juego no respondía correctamente.
- **Solución aplicada:** se implementaron cálculos básicos de comparación de coordenadas (X, Y) en lugar de algoritmos más complejos, lo que permitió un control eficiente y rápido.
- **Visión del resultado:** los juegos detectaron colisiones de manera fluida, con un impacto positivo en la jugabilidad.

Solución 2: Uso de layouts responsivos

- **Visión del problema:** la interfaz gráfica no se adaptaba a todas las pantallas.
- **Solución aplicada:** se usaron *ConstraintLayout* y porcentajes relativos en lugar de medidas fijas, lo que permitió escalabilidad en diferentes dispositivos.
- **Visión del resultado:** los juegos se visualizaron correctamente en celulares de distintos tamaños, manteniendo su jugabilidad.

Solución 3: Generación de números aleatorios controlada

- **Visión del problema:** en *Blackjack* o *Ruleta*, el azar podía repetirse o romper las reglas del juego.
- **Solución aplicada:** se implementaron funciones de aleatoriedad con rangos controlados, evitando repeticiones mediante listas dinámicas.
- **Visión del resultado:** la experiencia de juego se volvió más realista y confiable, sin resultados "trucados".

Solución 4: Validación de jugadas y estados

- **Visión del problema:** en *Solitario* y *Sudoku*, era posible realizar jugadas inválidas que rompían el flujo del juego.
- **Solución aplicada:** se añadieron verificaciones previas a cada acción del jugador, rechazando jugadas no válidas antes de ejecutarlas.
- **Visión del resultado:** los juegos mantuvieron su lógica sin errores, lo que garantizó estabilidad y una experiencia coherente.

Testimonios:

"El Blackjack está buenísimo, me recordó a cuando juego cartas con mis primos en reuniones. Lo mejor es que no se necesita internet, así en la uni mientras espero entre clases puedo echarme unas rondas. Eso sí, estaría genial que hubiera un modo donde uno pueda practicar con reglas más avanzadas."
– **Luis Ramírez, 21 años, estudiante de Administración**

2.

"El Brick Breaker se siente clásico, me atrapó desde el primer nivel. En mi celular corre súper fluido y los colores están muy vivos. Aunque estaría padre que hubiera distintos niveles de dificultad, porque al rato se vuelve fácil."
– **Andrea Torres, 20 años, estudiante de Diseño Gráfico**

3.

"Clicks123 me hizo reír bastante, es adictivo darle clics y ver cómo suben los puntos. Mis amigos ya lo descargaron también porque competimos a ver quién llega más rápido. Está sencillo, pero justo por eso no te cansa."
– **Sofía Hernández, 19 años, estudiante de Psicología**

4.

"Pac-Man fue mi favorito, literal me transportó a cuando veía a mi hermano jugar en maquinitas. Aquí no se traba y se escucha súper bien el sonido clásico. Sólo que estaría bueno tener más mapas o retos extra."
– **Carlos Medina, 23 años, estudiante de Ingeniería en Sistemas**

5.

"La Ruleta está divertida, siento que la hicieron muy realista. Me gusta porque aunque no sea con dinero, se siente emocionante. A veces las animaciones tardan un poquito

en cargar, pero nada que arruine la experiencia."
– **Daniela Ríos, 24 años, estudiante de Comunicación**

6.

"Snake fue el que más nostalgia me dio, mi primer celular tenía ese juego. Aquí se ve mucho más colorido y fácil de controlar. Aunque a veces me cuesta mover la serpiente rápido porque el espacio en pantalla es reducido."
– **Alejandro Vargas, 22 años, estudiante de Arquitectura**

7.

"Me sorprendió que trajeran Solitario, pensé que ya nadie lo jugaba. La interfaz está muy clara, y mi mamá también lo probó y se le hizo fácil. Lo único es que estaría bien poder cambiar los diseños de las cartas."
– **Paola Jiménez, 20 años, estudiante de Medicina**

8.

"Sudoku me encantó porque lo uso para despejarme entre tareas. Tiene distintos niveles y eso me ayuda a practicar. Lo único es que después de un rato me gustaría que hubiera un cronómetro o rankings para hacerlo más retador."
– **Javier López, 25 años, recién egresado de Economía**

9.

"Me gusta mucho que en una sola app tengas de todo: Pac-Man, Snake, cartas, la Ruleta. Ya no tengo que estar bajando varias apps. Aunque la pantalla de inicio podría verse más moderna, siento que es un área a mejorar."
– **Mariana Castillo, 18 años, estudiante de Preparatoria**

10.

"El catálogo está bien variado y siento que cada juego está optimizado, ninguno me ha fallado. Lo que más me gustó es que no ocupa tantos datos ni espacio en mi celular. Solo pediría que en la próxima versión agreguen logros o recompensas para hacerlo

más competitivo."
– **Fernando Cruz, 19 años, estudiante de Derecho**

Testimonios Escalabilidad

"La neta me gustan un chorro los juegos, sobre todo el Snake y el Brick Breaker, me entretienen cuando voy en el camión. Pero estaría padrísimo que pusieran logros o retos diarios, así uno se pica más y no solo juega por jugar."
– **Claudia Ramírez, 22 años, estudiante de Ingeniería Industrial**

2.

"Yo juego mucho el Pac-Man y el Sudoku, y la verdad sí me la paso bien. Aunque estaría genial que hubiera un ranking en línea para ver quién saca más puntos, así siento que competiría con otros y no solo conmigo mismo."
– **Miguel Torres, 24 años, estudiante de Ciencias de la Computación**

3.

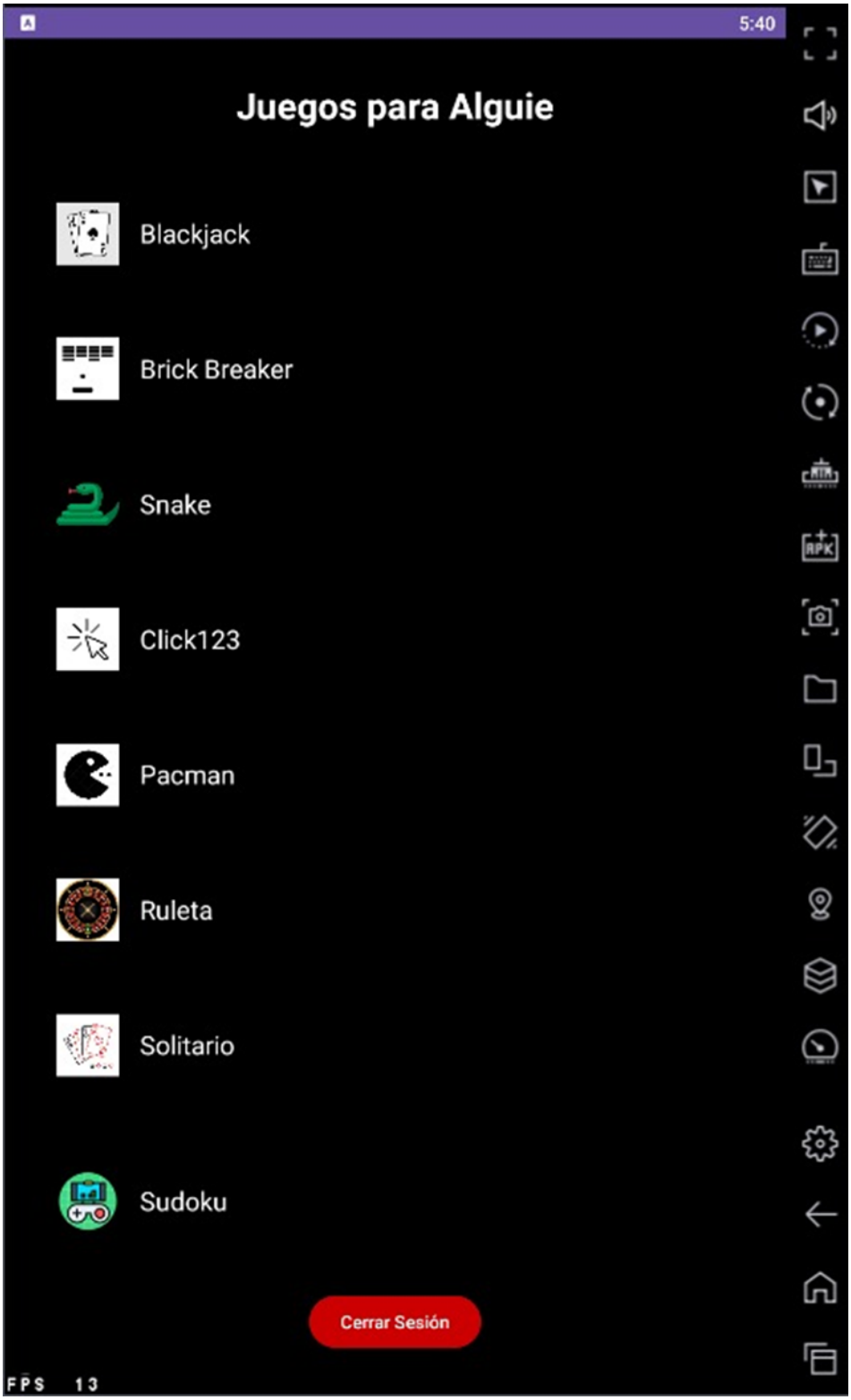
"Lo que más me gusta es que no ocupa internet, me salva en las clases aburridas jajaja. Pero creo que si agregaran un par de juegos nuevos cada mes, tipo Tetris o alguno de disparos retro, estaría todavía más completo."
– **Valeria Gómez, 19 años, estudiante de Medicina**

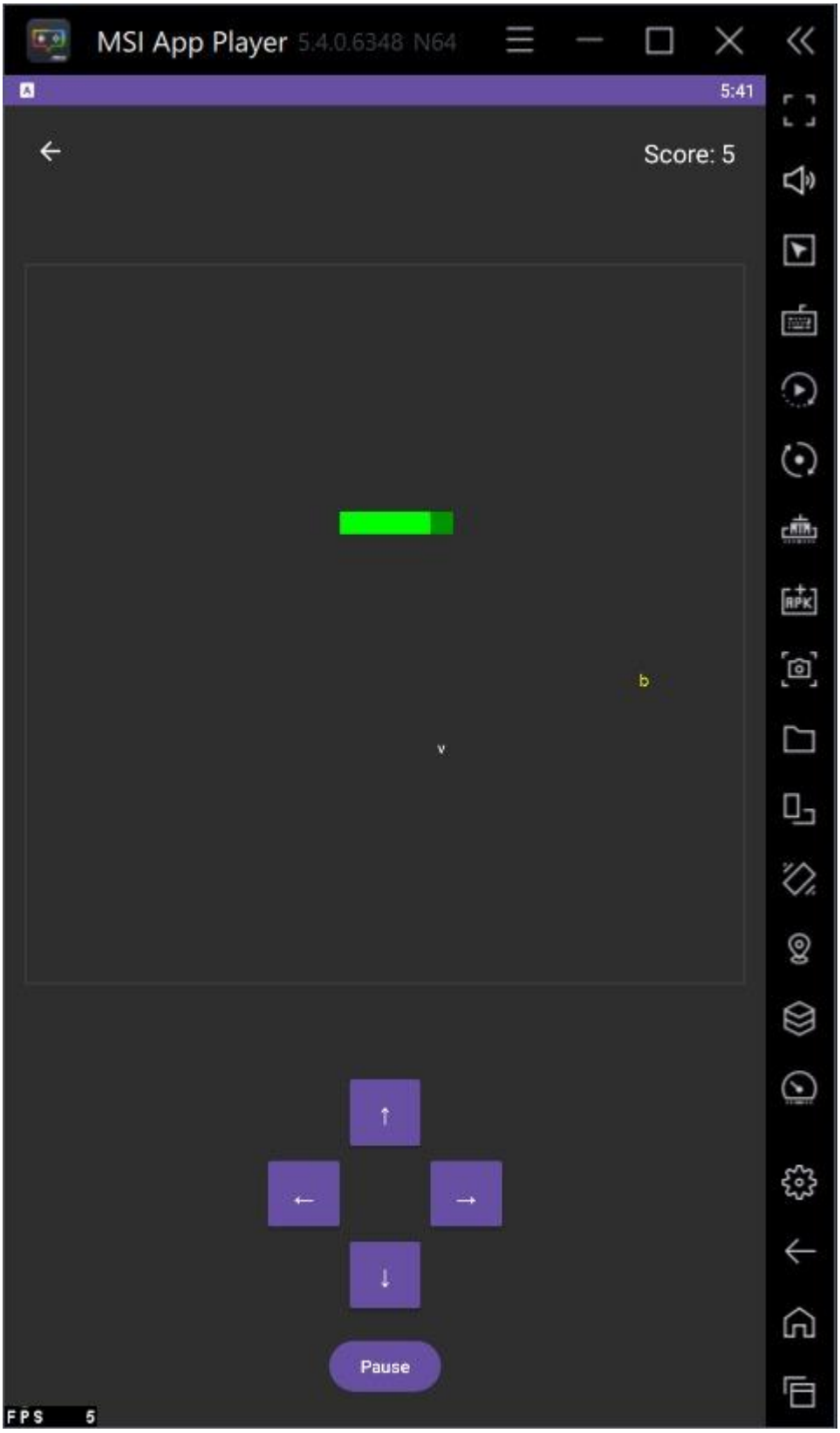
4.

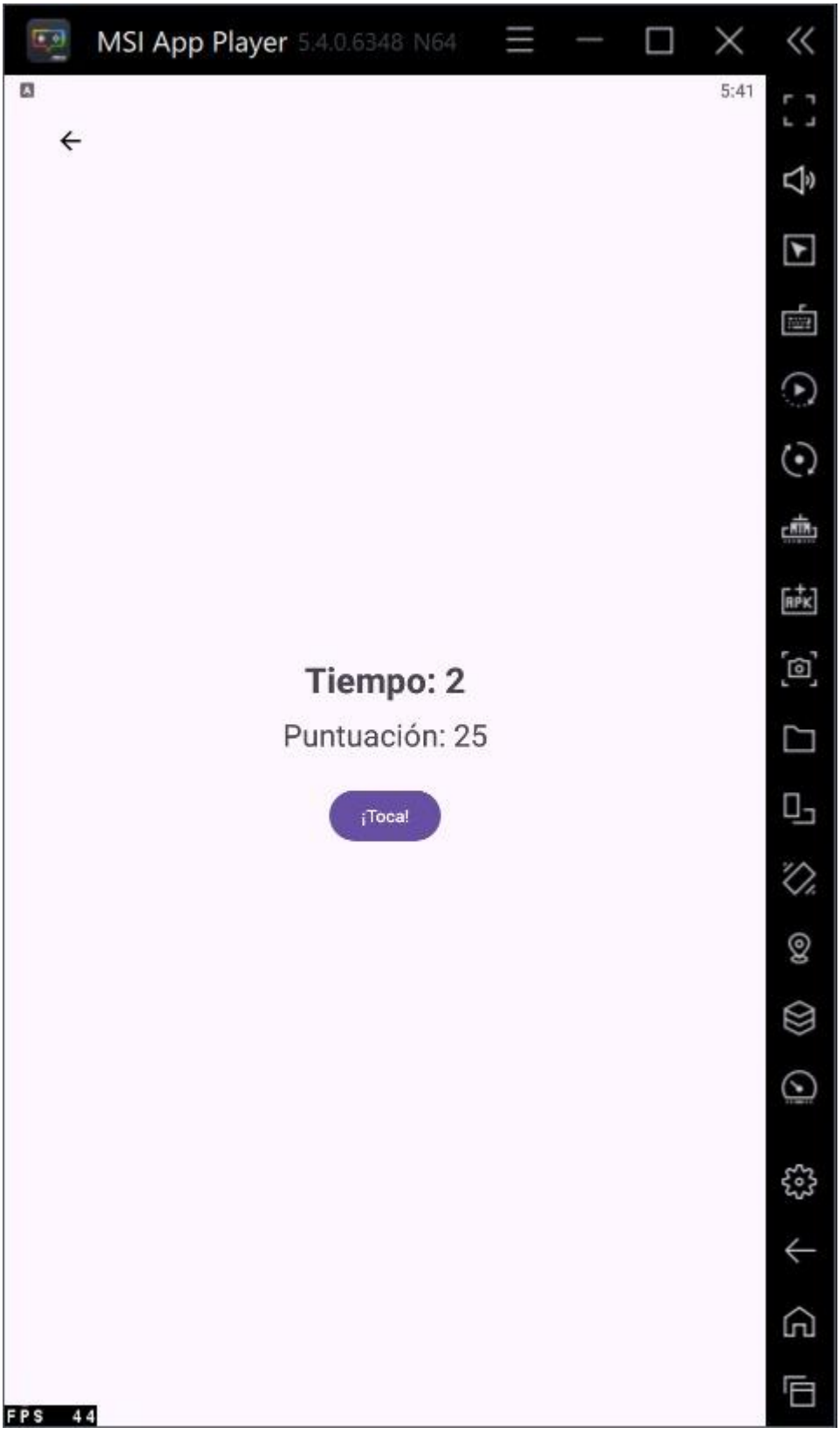
"Algo que estaría muy chido es que cada quien tuviera un perfil donde se guarden las estadísticas, los récords y hasta ver cuánto tiempo juegas cada título. Sería como tener tu propio historial y darle más sentido a lo que juegas."
– **Juan Pablo Martínez, 21 años, estudiante de Derecho**

5.

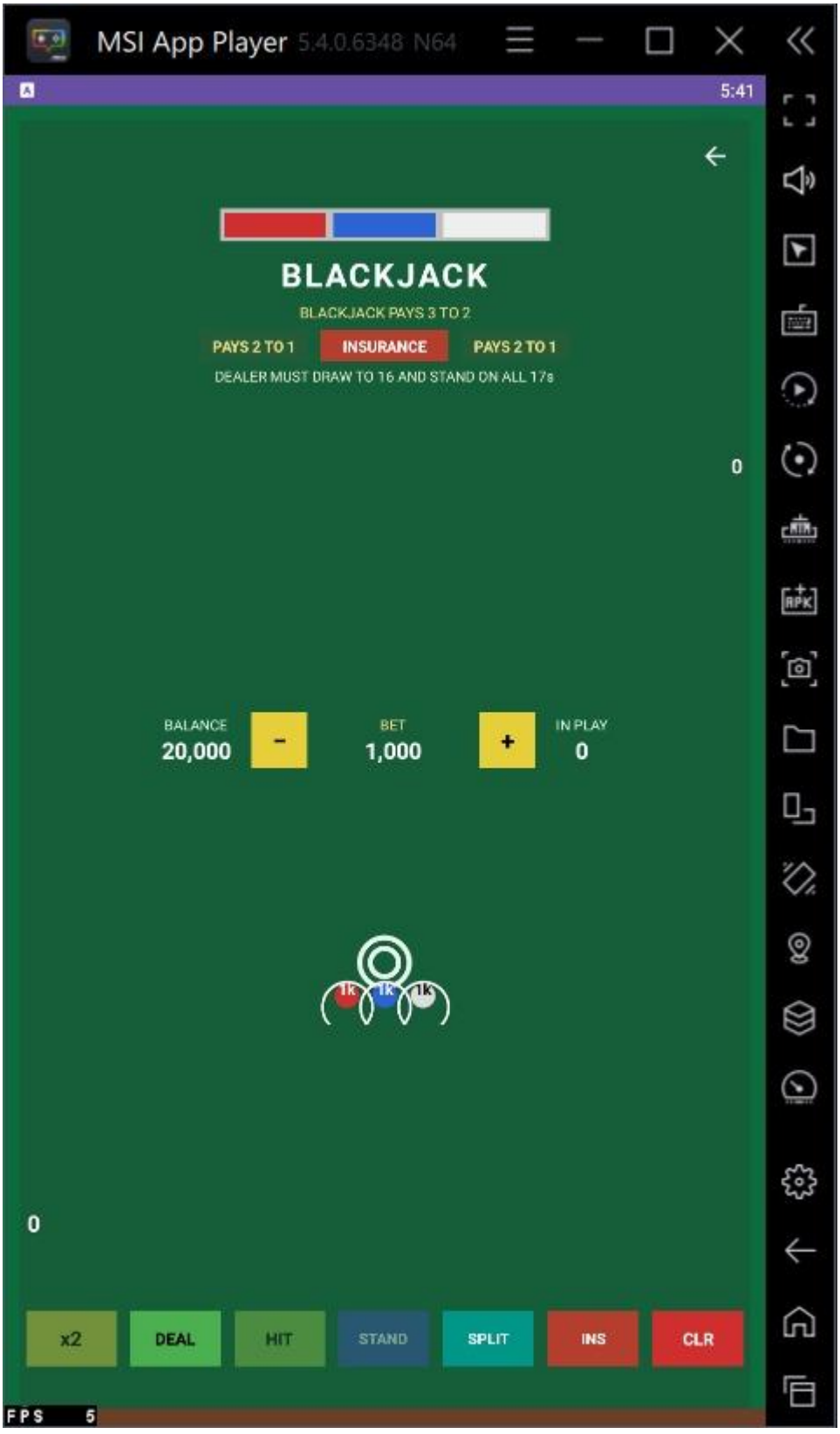
"Yo casi siempre juego la Ruleta y el Blackjack con mis amigos, y lo pasamos entre varios. Si en el futuro se pudiera jugar en modo multijugador, aunque sea por Bluetooth o pasando el cel, sería muchísimo más divertido."
– **Ana Sofía Delgado, 20 años, estudiante de Comunicación**

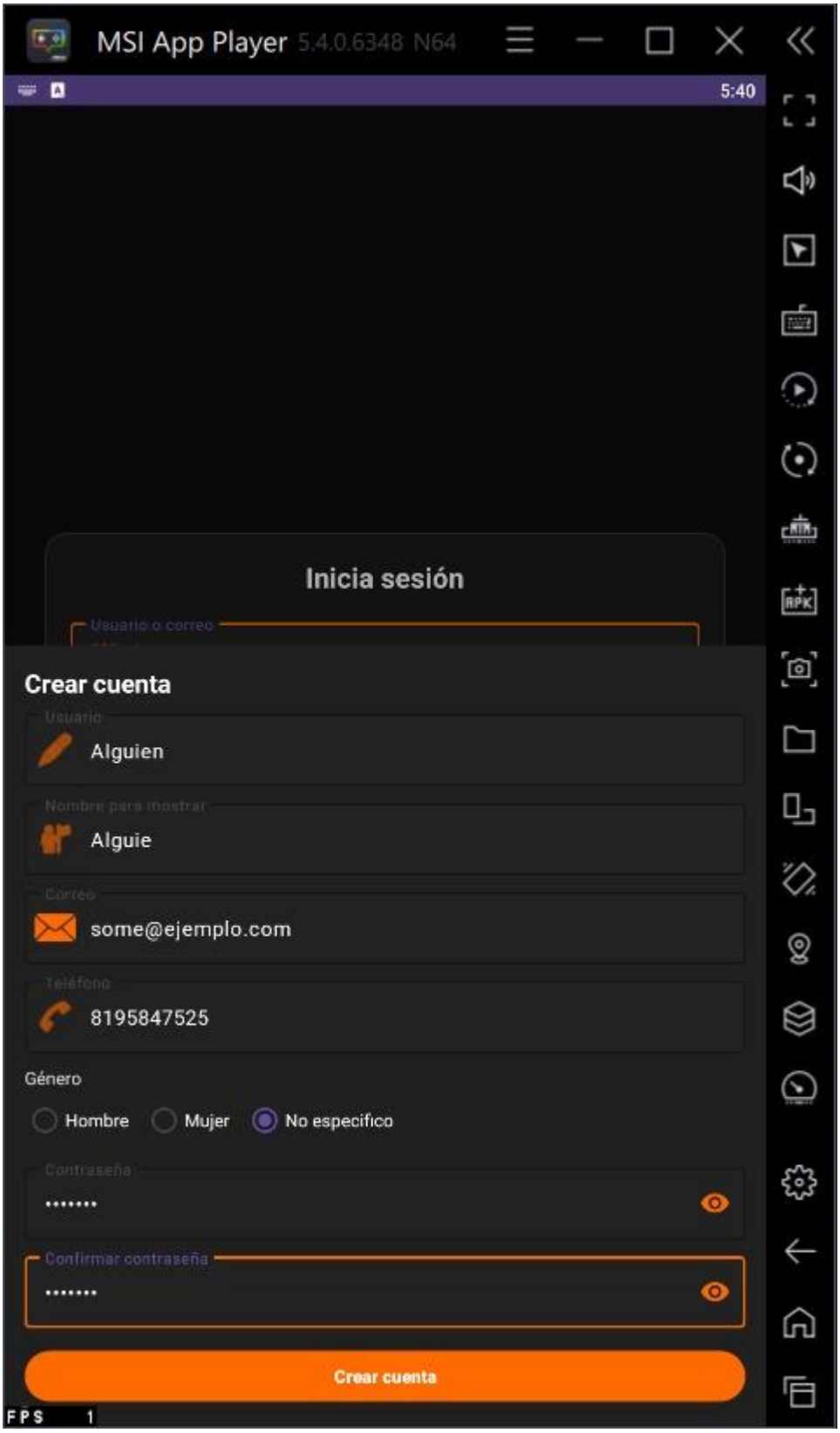

5:40
Juegos para Alguie
Blackjack
Brick Breaker
Snake
Click123
Pacman
Ruleta
Solitario
Sudoku
Cerrar Sesión
FPS 13

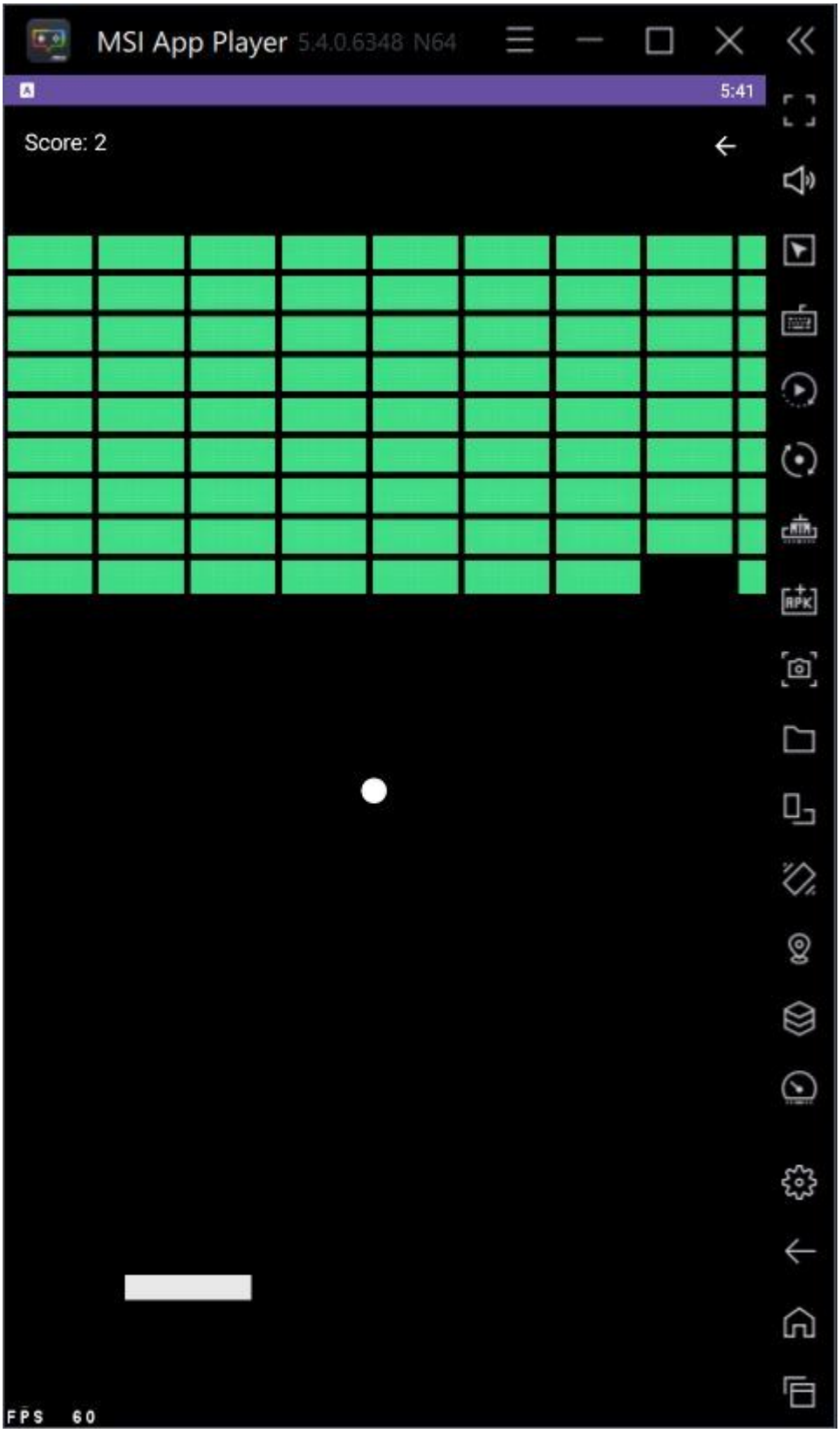

MSI App Player 5.4.0.6348 N64
5:41
Score: 5
b
v
Pause
FPS 5

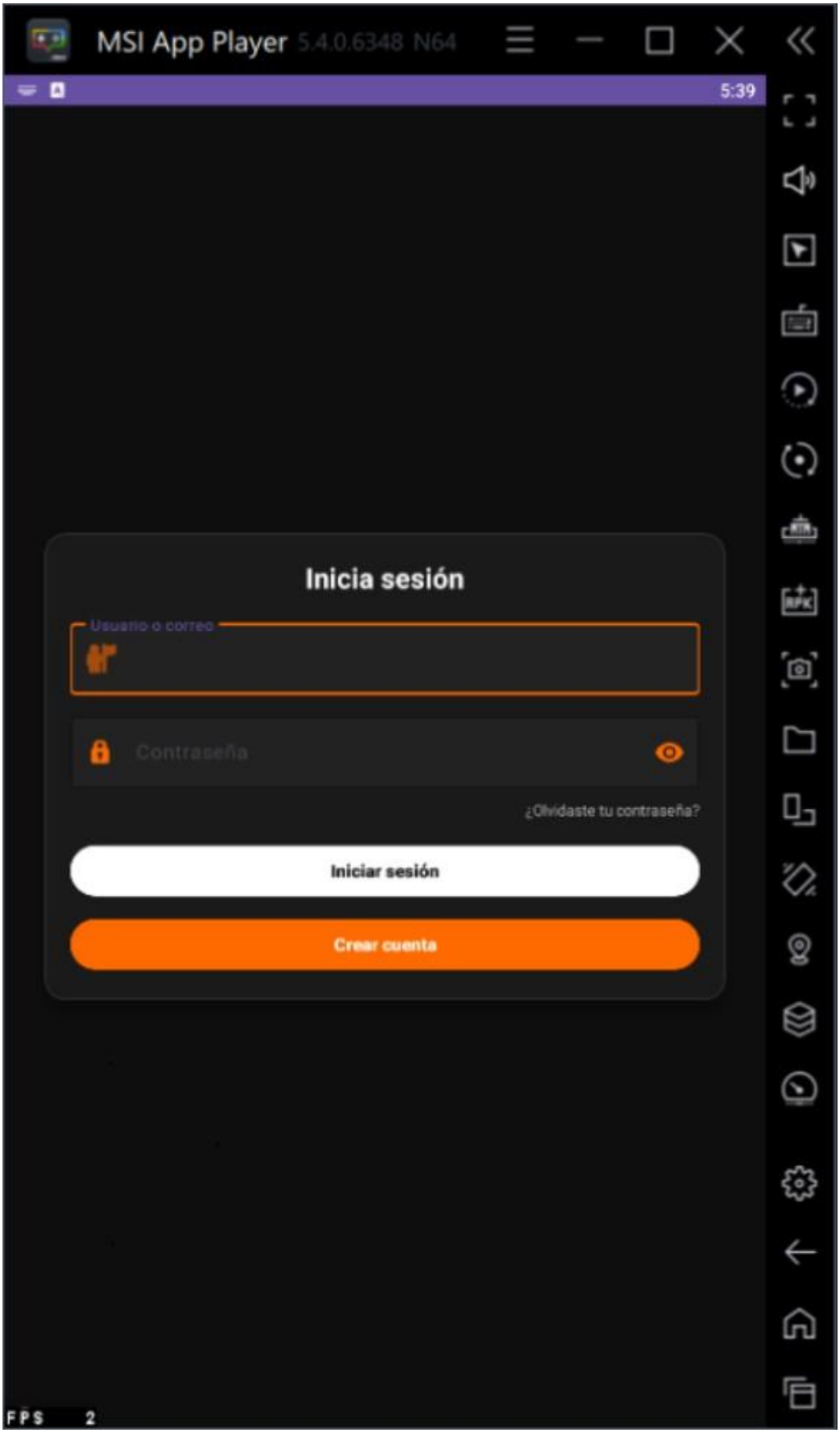

MSI App Player 5.4.0.6348 N64
5:41
Tiempo: 2
Puntuación: 25
¡Toca!
FPS 44

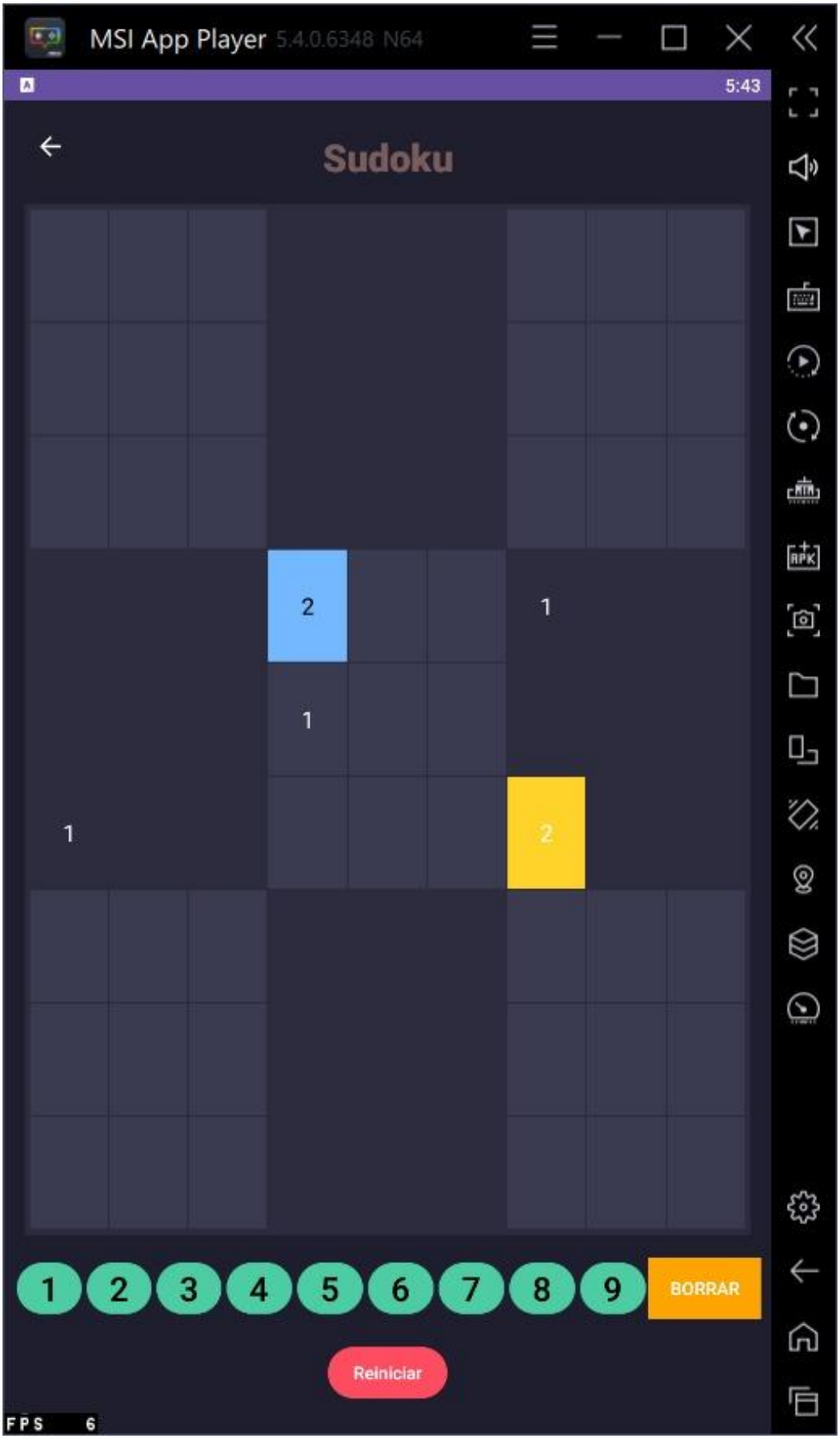

MSI App Player 5.4.0.6348 N64
5:42
PAC-MAN
SCORE 000240 HIGH 000000
START
RESET
FPS 59

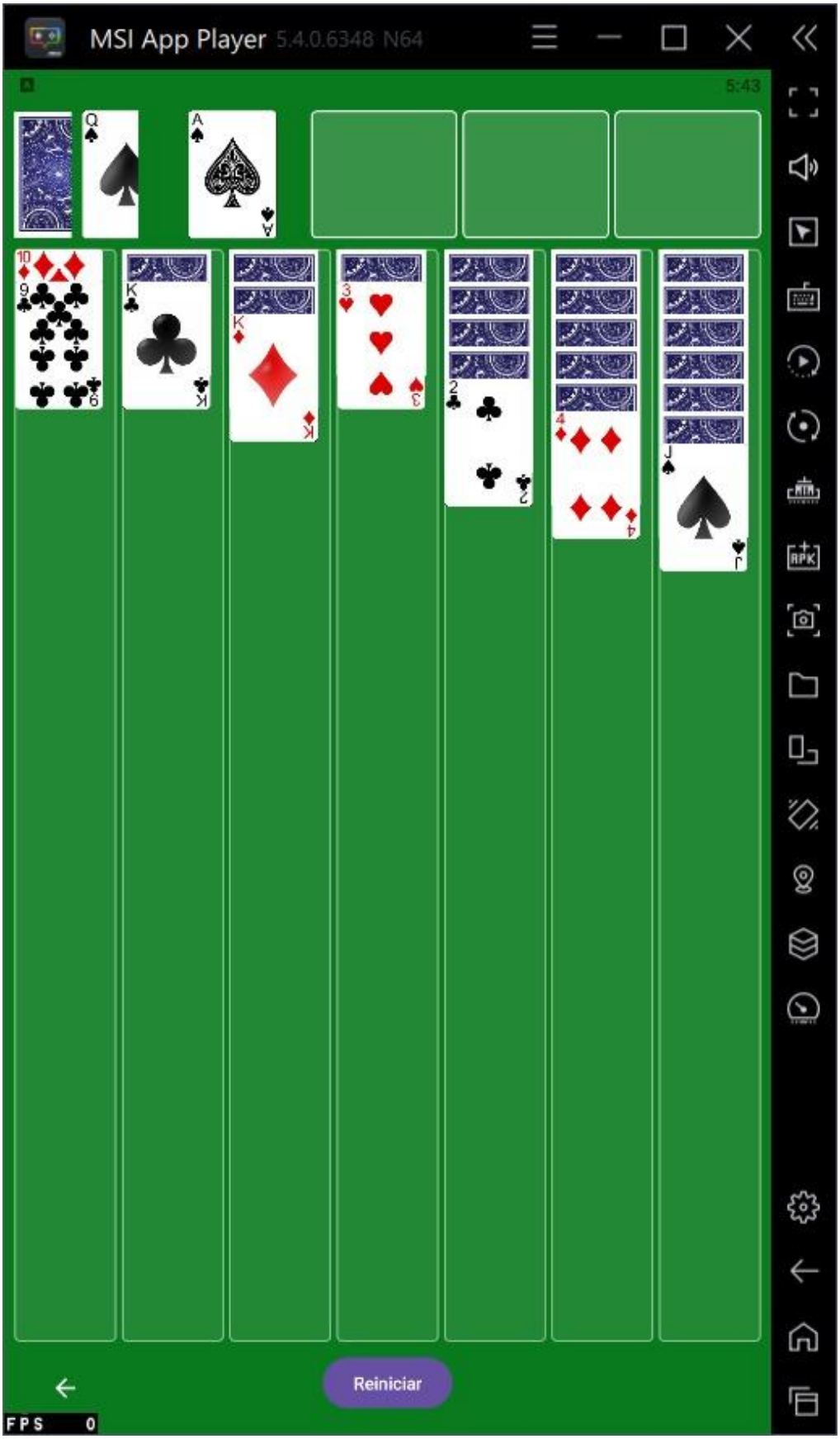

MSI App Player 5.4.0.6348 N64
5:41
BLACKJACK
BLACKJACK PAYS 3 TO 2
PAYS 2 TO 1
INSURANCE
PAYS 2 TO 1
DEALER MUST DRAW TO 16 AND STAND ON ALL 17s
0
BALANCE
20,000
-
BET
1,000
+
IN PLAY
0
0
x2
DEAL
HIT
STAND
SPLIT
INS
CLR
FPS 5

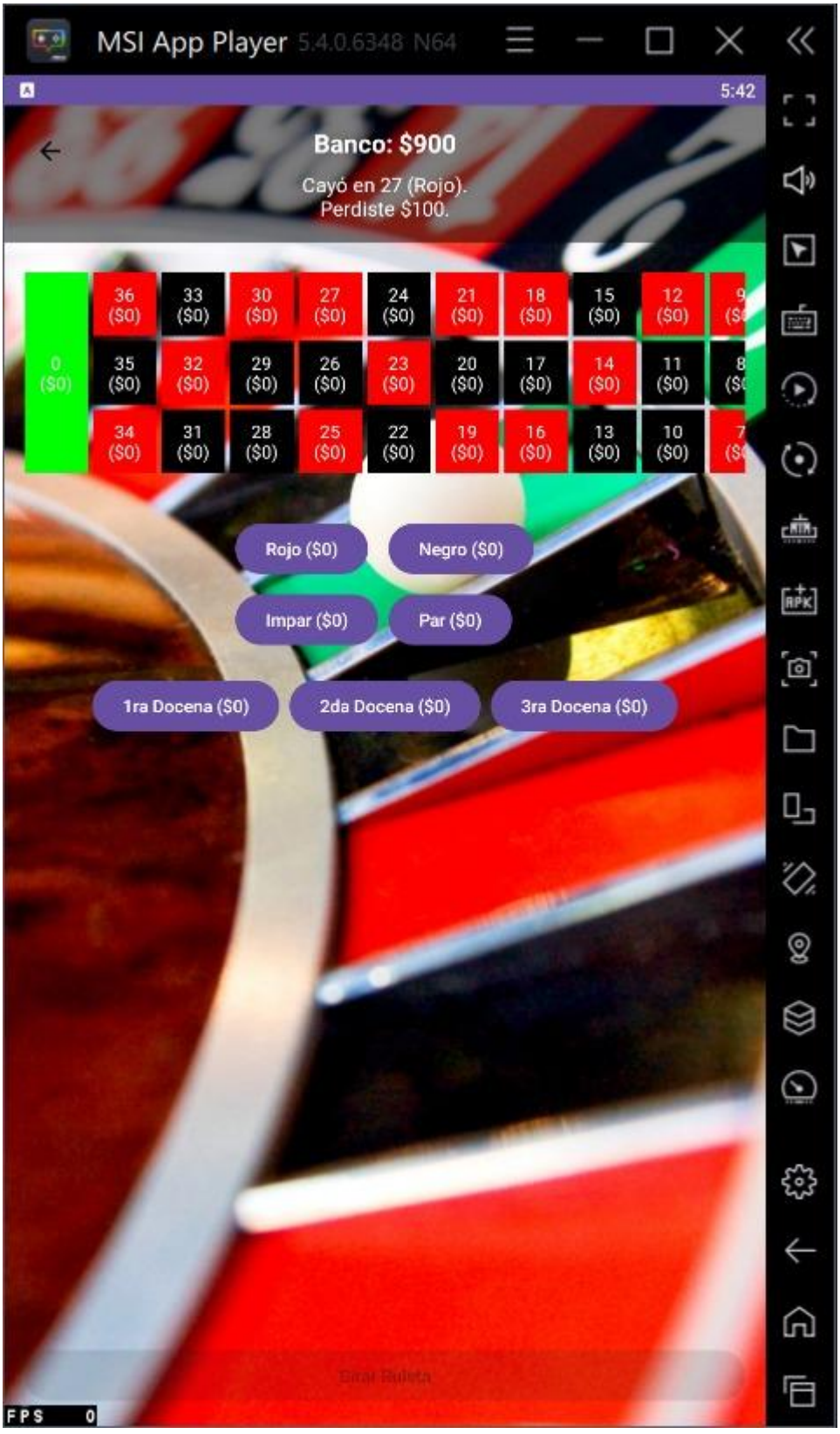

MSI App Player 5.4.0.6348 N64
5:40
Inicia sesión
Usuario o correo
Crear cuenta
Usuario
Alguien
Nombre para mostrar
Alguie
Correo
some@ejemplo.com
Teléfono
8195847525
Género
Hombre
Mujer
No especifico
Contraseña
Confirmar contraseña
Crear cuenta
FPS 1


MSI App Player 5.4.0.6348 N64
5:41
Score: 2
FPS 60


MSI App Player 5.4.0.6348 N64
5:39
Inicia sesión
Usuario o correo
Contraseña
¿Olvidaste tu contraseña?
Iniciar sesión
Crear cuenta
FPS 2


MSI App Player 5.4.0.6348 N64
5:43
Sudoku
2
1
1
1
2
1 2 3 4 5 6 7 8 9
BORRAR
Reiniciar
FPS 6


MSI App Player 5.4.0.6348 N64
5:43
Reiniciar
FPS 0


MSI App Player 5.4.0.6348 N64
5:42
Banco: $900
Cayó en 27 (Rojo).
Perdiste $100.
0 ($0)
36 ($0)
33 ($0)
30 ($0)
27 ($0)
24 ($0)
21 ($0)
18 ($0)
15 ($0)
12 ($0)
35 ($0)
32 ($0)
29 ($0)
26 ($0)
23 ($0)
20 ($0)
17 ($0)
14 ($0)
11 ($0)
34 ($0)
31 ($0)
28 ($0)
25 ($0)
22 ($0)
19 ($0)
16 ($0)
13 ($0)
10 ($0)
Rojo ($0)
Negro ($0)
Impar ($0)
Par ($0)
1ra Docena ($0)
2da Docena ($0)
3ra Docena ($0)
FPS 0

Conclusión

Al finalizar el desarrollo de nuestra aplicación de juegos, logramos crear un entorno completo, atractivo y funcional que combina entretenimiento, accesibilidad y rendimiento. La separación de responsabilidades entre la interfaz, la lógica y la persistencia permitió que cada juego funcione de manera independiente, manteniendo consistencia y fluidez, desde Blackjack y Pac-Man hasta Brick Breaker, Snake, Ruleta, Clicks123, Sudoku y Solitaire.

La experiencia del usuario fue nuestra prioridad: diseñamos pantallas claras, controles intuitivos, catálogo organizado y mecánicas de juego que cualquiera puede entender y disfrutar, incluso sin conocimientos técnicos. Además, implementamos mecanismos para guardar el progreso, mantener sesiones activas y facilitar la navegación entre juegos, garantizando que la diversión no se interrumpa y que los usuarios puedan explorar todos los títulos disponibles sin complicaciones.

También nos enfocamos en la escalabilidad y seguridad: la arquitectura modular permite futuras actualizaciones, la adición de nuevos juegos o funcionalidades, y la integración con servicios externos sin afectar la experiencia de quienes ya usan la aplicación. Por otro lado, los datos de usuario se manejan de forma segura, preparando el camino para una posible conectividad con backend remoto en futuras versiones.

En resumen, logramos desarrollar una aplicación robusta, intuitiva y adaptable, que combina diversión, rendimiento y facilidad de uso. Nuestra app no solo ofrece un repertorio variado de juegos clásicos y arcade, sino que también establece una base sólida para crecer, evolucionar y seguir ofreciendo entretenimiento de calidad a nuestros usuarios.