Entrega 1

Integrantes: Amparo Johnson y Sofía Weiffenbach

1. Descripción general del juego y su propósito

Qué es el juego

"Escaleras y Serpientes" es un juego de mesa digital multijugador basado en el clásico juego de tablero. Se trata de una adaptación web que mantiene la esencia del juego tradicional mientras incorpora elementos modernos como cartas especiales, monedas virtuales y eventos aleatorios. El juego se desarrolla en un tablero numerado del 1 al 100, donde los jugadores avanzan por turnos en función del resultado de lanzar un dado, con el objetivo de llegar primero a la casilla final. El tablero está interconectado por escaleras que permiten subir rápidamente y serpientes que hacen retroceder, añadiendo un elemento estratégico y de azar al juego. La motivación para la creación de esta aplicación en particular es el volver a encantar al público con este juego nostálgico tan familiar, dándole una mirada innovadora y tecnológica.

Objetivo del jugador

El objetivo principal es ser el primer jugador en alcanzar exactamente la casilla número 100. Para lograrlo, los jugadores deben:

- Avanzar estratégicamente por el tablero lanzando el dado en su turno
- Aprovechar las escaleras para subir rápidamente a casillas superiores
- Evitar caer en serpientes que los hacen retroceder
- Utilizar sabiamente las cartas especiales adquiridas durante el juego
- Administrar eficientemente las monedas virtuales para comprar cartas ventajosas
- Planificar sus movimientos considerando el mecanismo de "rebote" cuando están cerca de la casilla final

Mecánica general

El juego funciona mediante un sistema por turnos donde cada jugador tiene oportunidad de realizar acciones específicas:

- **Lanzamiento de dado**: En su turno, el jugador debe lanzar un dado virtual de 6 caras que determina cuántas casillas avanzará.
- **Sistema de movimiento**: El jugador avanza automáticamente el número de casillas indicado por el dado. Si cae en una casilla con una escalera, asciende inmediatamente a una casilla superior; si cae en una con serpiente, desciende a una inferior.
- **Sistema de rebote**: Cuando un jugador está cerca del final (casilla 100) y obtiene un valor en el dado que lo haría pasar de la casilla 100, "rebota" hacia atrás. Por ejemplo, si está en la casilla 98 y obtiene un 5, avanzará hasta 100 y rebotará 3 casillas hacia atrás, quedando en la 97.

- **Economía de juego**: Los jugadores pueden ganar monedas virtuales al caer en casillas especiales, que pueden utilizar para comprar cartas con habilidades especiales.
- **Sistema de cartas**: Existen dos formas de obtener cartas:
    - Comprándolas con monedas durante su turno
    - Automáticamente cada 3 turnos
- **Casillas especiales**: Además de las escaleras y serpientes, existen casillas que activan eventos positivos o negativos, como ganar monedas o sufrir penalizaciones.
- **Condición de victoria**: El primer jugador que llegue exactamente a la casilla 100 gana la partida.

Roles involucrados

Los roles involucrados en el funcionamiento del juego son: Jugador (Que se une o es dueño de la partida), Administrador, Invitado (usuario no registrado). Las descripciones y funcionalidades de cada rol se detallan más adelante.

2. **Reglas del juego y comportamiento del servidor**

Mecánica del juego:
1. El juego es por turnos (uno a la vez, en orden circular).

2. En su turno, un jugador puede:

- Lanzar un dado de 6 caras.
- Comprar cartas (si tiene monedas disponibles)
- Usar una carta (si tiene una disponible).

3. El jugador avanza tantas casillas como el valor del dado.

4. Si cae en una:

- **Escalera**: sube a una casilla superior conectada.
- **Serpiente**: baja a una casilla inferior conectada.
- **Casilla especial**: puede activar un evento positivo o negativo. Por ejemplo, ganar monedas.

5. Si un jugador cae exactamente en la casilla 100, gana.

- Si el valor del dado hace que se pase, rebota hacia atrás (ej. está en 98, lanza 5 → rebota a 97).

Eventos especiales:

Además de la posibilidad de comprar cartas, cada 3 turnos, un jugador obtiene una **carta especial**, que puede ser:

- Tirar 2 dados

- Intercambiar posición con otro jugador
- Saltarse la próxima serpiente
- Volver un turno a otro jugador

3. Tipos de usuario y casos de uso:

- **Jugador:** puede crear y unirse a partidas, lanzar dado, comprar y usar cartas.

  Descripción: Usuario principal del sistema, puede crear y jugar partidas, lanzar dados, comprar y usar cartas, etc.

  Funcionalidades:

  - Crear una nueva partida y ser el "dueño" de ella.
  - Unirse a una partida activa (si hay cupo).
  - Ver el estado de sus partidas activas o pasadas.
  - Ver el tablero y su posición.
  - Lanzar el dado cuando sea su turno.
  - Comprar cartas especiales (si tiene suficientes monedas)
  - Usar cartas especiales (si tiene).
  - Abandonar una partida (se marcará como inactiva para ese usuario).
  - Visualizar historial de acciones.

- **Administrador:** puede eliminar partidas, usuarios, y ver auditorías.

  Funcionalidades:

  - Ver todas las partidas, incluso privadas.
  - Eliminar usuarios o partidas.
  - Ver logs completos del sistema.
  - Suspender jugadores por mal comportamiento.
  - Agregar o quitar moderadores.
  - Ver estadísticas globales (cantidad de partidas, usuarios activos, promedio de duración, etc.)

- **Invitado (no registrado):** solo puede visualizar partidas públicas.

Descripción: Puede navegar por la página principal y visualizar partidas públicas, pero no puede participar directamente en ninguna.

Funcionalidades:

- Ver landing page y reglas del juego.
- Ver una lista de partidas públicas en curso.
- Observar el desarrollo de una partida pública (modo "espectador", sin interacción).

- Acceder al registro o login.

4. **Registro de usuario y navegación**

Para participar activamente en el juego, cada usuario debe registrarse mediante un formulario de creación de cuenta. La página de registro está diseñada para ser clara, accesible y segura, considerando una buena experiencia de usuario (UX).

El formulario de registro solicita la siguiente información:

- **Nombre de usuario**: debe ser único, ya que identifica al jugador dentro del sistema.
- **Correo electrónico**: necesario para asociar la cuenta a un canal de contacto y para recuperación de cuenta.
- **Contraseña**: enmascarada, con una longitud mínima recomendada de 8 caracteres.
- **Confirmación de contraseña**: para evitar errores de tipeo.
- **Avatar** (opcional): permite al usuario subir una imagen personalizada.

Una vez ingresados los datos, el formulario se envía al servidor mediante una petición POST. El servidor valida la información y responde con un mensaje de éxito o con los errores correspondientes (por ejemplo, si el usuario ya existe o las contraseñas no coinciden). Tras un registro exitoso, el usuario es redirigido al login o, si se desea, accede automáticamente a su cuenta y es redirigido al **lobby de juego**.

Una vez registrado e iniciado sesión, el usuario accede a una interfaz intuitiva que le permite realizar las siguientes acciones:

- Ver su perfil personal (nombre, avatar, historial de partidas).
- Crear una nueva partida o unirse a una existente.
- Ver partidas públicas en curso.
- Acceder a la partida en la que está participando.
- Cerrar sesión.

5. **Mockup de Landing page**

6. **Mockup de registro de usuario**

7. **Registro e integración a partidas**

- **Creación y unión a partidas**

Una vez que un usuario ha iniciado sesión, puede optar por **crear una nueva partida** o **unirse a una partida existente**. Para crear una partida, el usuario debe asignarle un nombre y elegir su configuración básica (por ejemplo, si será pública o privada, cantidad máxima de jugadores y modo de juego si existen variantes). Quien crea la partida es automáticamente designado como su **dueño**.

Para unirse a una partida, los jugadores registrados pueden navegar por una lista de partidas activas, filtradas por estado (públicas, privadas con invitación, en espera, en curso). En el

caso de partidas privadas, se requiere un código de invitación o aprobación del dueño para ingresar.

- **Salas de espera**

Las partidas tienen un estado inicial de **"en espera"** hasta que el mínimo de jugadores requerido se haya unido (al menos 2). Durante este estado, los jugadores pueden:

- Confirmar su participación
- Abandonar la sala antes de iniciar

Una vez que se alcanza el mínimo necesario, el dueño de la partida puede iniciar la partida manualmente, o bien esta se iniciará automáticamente si se completa el número máximo de jugadores y se cumple el tiempo de espera (ver siguiente sección).

- **Tiempo límite**

Las partidas tienen un **tiempo límite de espera** desde su creación para iniciarse. Este límite será configurable (por ejemplo, 10 minutos por defecto). Si transcurrido ese tiempo no se reúnen los jugadores necesarios, la partida:

- Se elimina automáticamente si solo hay un jugador.
- Se reconfigura como pública o se mantiene en estado latente si hay al menos dos jugadores.

8. **Navegación durante el juego / Pantallas y acciones**

Durante una partida, el jugador accede a una vista central compuesta por los siguientes elementos:

1. **Tablero principal**:
    - Representado como una grilla de 10x10 casillas numeradas del 1 al 100.
    - Cada casilla puede contener íconos que representan **escaleras**, **serpientes**, **eventos especiales** y/o **jugadores**.
    - Al caer en una casilla especial, se realizará la acción automáticamente, ya sea escalera se subirá lo necesario, serpiente ocurrirá lo mismo y evento especial será un pop up que explica lo ocurrido y luego ocurre.
2. **Fichas de jugadores**:
    - Cada jugador está representado por una ficha o avatar que se desplaza visualmente por el tablero.
    - Cada jugador será asignado un color al comenzar el juego para que sea identificable.
3. **Panel lateral de información**:

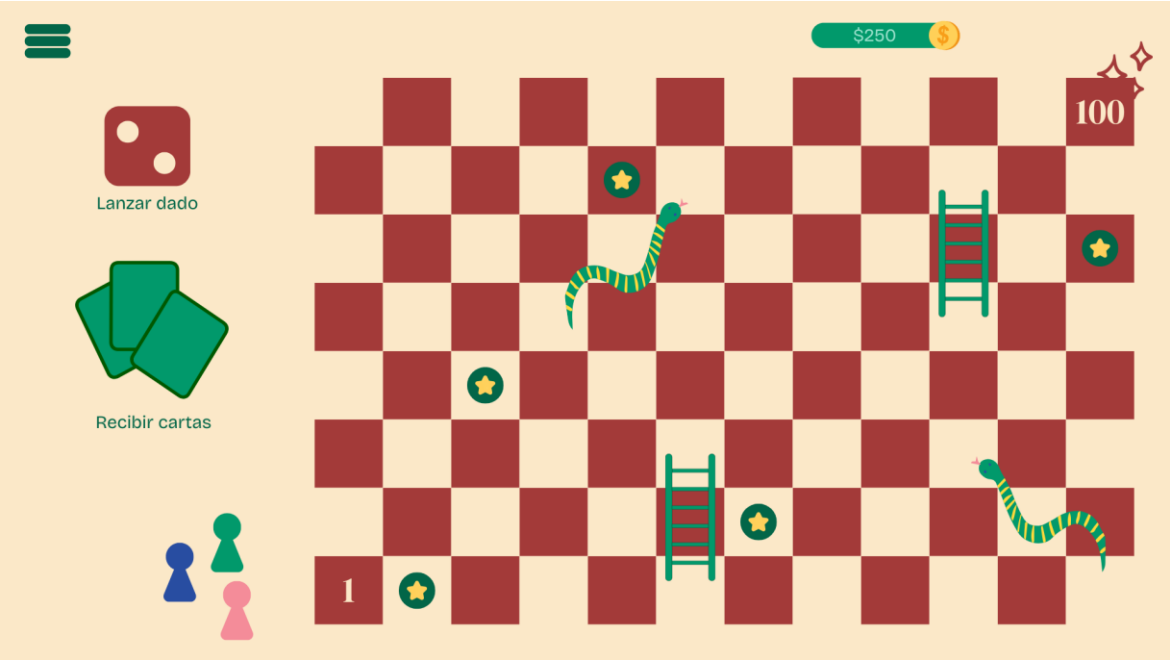
- Muestra: jugador actual, resultado del dado, cartas disponibles (si las hay) y la cantidad de monedas disponibles.
- También incluye el icono del menú para acceder al menú lateral en caso de ser necesario.
- Los siguientes botones también se encontraran disponibles:
  - **“Lanzar dado”** (activo solo si es el turno del jugador).
  - **“Usar carta”** (si el jugador tiene una).
  - **“Comprar carta”** (si el jugador tiene monedas suficientes)

Las interacciones en el tablero serán automáticas, las acciones en manos del jugador serán, usar carta, comprar carta o visualizar cartas y lanzar el dado. Como se explica en mas detalle a continuación:

- **Lanzamiento de dado**: se realiza haciendo clic en el botón correspondiente. El dado se anima y el resultado se muestra visualmente.
- **Movimiento de fichas**: automático tras el lanzamiento del dado, con animación suave hasta la nueva posición. No requiere intervención manual.
- **Cartas y eventos**:
  - Si el jugador posee cartas especiales, puede hacer clic en una de ellas para activarla.
  - Algunas cartas podrían requerir seleccionar a otro jugador (se hace clic sobre su ficha).
- **Feedback visual inmediato**:
  - Cuando el jugador sube por una escalera o baja por un tobogán, se destaca la trayectoria con una animación.
  - Mensajes emergentes informan cada acción o evento que ocurre en el turno.

9. Visualización del tablero o mapa

A continuación, se muestra un mockup de cómo se verá el tablero de juego, la información será publica excepto la cantidad de monedas disponibles o al hacer click en las cartas, en donde aparecerán las cartas disponibles para un jugador, junto con la posibilidad de usarlas o comprar más.



10. Elementos aleatorios

Los elementos fortuitos del juego son los siguientes:

Lanzamiento del dado

- Cada turno comienza con el lanzamiento de un dado virtual de 6 caras, cuyo resultado determina cuántas casillas avanza el jugador.
- Así se introduce azar en el desplazamiento, manteniendo la esencia del juego tradicional y evitando que los resultados sean totalmente previsibles.

Escaleras y Serpientes

- Las casillas con escaleras y serpientes representan eventos aleatorios predefinidos en el tablero.
- Aunque sus ubicaciones son fijas, su impacto depende totalmente del resultado del dado, haciendo que un mismo valor pueda tener consecuencias radicalmente distintas dependiendo del contexto del tablero.
    - **Escaleras**: permiten al jugador ascender instantáneamente a una casilla superior.
    - **Serpientes**: obligan al jugador a retroceder a una casilla inferior.

Casillas especiales

- Algunas casillas activan **eventos fortuitos** al ser pisadas, los cuales pueden tener efectos positivos o negativos. Estos eventos son definidos aleatoriamente por el servidor entre un conjunto preestablecido de posibilidades. Ejemplos:
    - Ganar monedas virtuales
    - Perder monedas

- Perder el próximo turno
- Robar una carta aleatoria a otro jugador
- Recibir una carta gratis

Cartas especiales

- Cada jugador recibe automáticamente una carta **cada 3 turnos**, independientemente de su posición.
- Estas cartas, entregadas de forma aleatoria desde un conjunto limitado, ofrecen efectos únicos y estratégicos que pueden cambiar el rumbo de la partida:
  - Tirar 2 dados en lugar de 1
  - Intercambiar posición con otro jugador
  - Anular una serpiente en el próximo turno
  - Hacer retroceder a otro jugador

Sistema de monedas y compra de cartas

- Algunos eventos o casillas otorgan **monedas**, que el jugador puede usar para comprar cartas especiales.
- La disponibilidad de cartas para compra también es aleatoria dentro del conjunto total de cartas del juego.
- Esto añade un componente de incertidumbre y estrategia en la administración de recursos.

11. Protocolo de comunicación cliente-servidor

El protocolo de comunicación entre el cliente y el servidor se basa en una arquitectura REST con intercambio de mensajes en formato JSON a través de WebSockets para garantizar actualizaciones en tiempo real durante el juego.

I. **Mensajes del Cliente al Servidor:**

11. Protocolo de comunicación cliente-servidor

El juego implementa un protocolo de comunicación cliente-servidor bidireccional para garantizar que todos los jugadores tengan una experiencia sincronizada. Se adjunta el archivo *cliente_servidor.json* para mostrar ejemplos de estructuras de datos JSON de la comunicación cliente-servidor. A continuación, se detallan los principales flujos de datos:

Desde el cliente al servidor:

1. **Registro e inicio de sesión**
   a. Datos de registro de nuevo usuario

b. Credenciales de inicio de sesión
c. Solicitudes de recuperación de contraseña

2. **Gestión de partidas**
    a. Creación de nueva partida (configuración, privacidad)
    b. Solicitud para unirse a una partida existente
    c. Envío de código de invitación para partidas privadas
    d. Confirmación de participación en sala de espera
    e. Solicitud para iniciar partida (solo dueño)
    f. Abandono de partida o sala de espera
3. **Acciones durante el juego**
    a. Solicitud de lanzamiento de dado
    b. Solicitud de compra de carta
    c. Uso de carta especial (incluyendo selección de jugador objetivo si es necesario)
    d. Mensajes de chat entre jugadores (opcional)
    e. Solicitud de historial de movimientos

II. **Desde el servidor al cliente:**

1. **Respuestas de autenticación**
    a. Confirmación de registro exitoso o errores
    b. Confirmación de inicio de sesión o errores
    c. Información de perfil del usuario
2. **Información de partidas**
    a. Lista de partidas disponibles (públicas y por invitación)
    b. Estado actual de la sala de espera (jugadores, configuración)
    c. Notificación de inicio de partida
    d. Notificación de fin de partida con resultados
    e. Actualizaciones de participantes (nuevos, abandonos)
3. **Actualización del estado del juego**
    a. Estado completo del tablero (posición de todos los jugadores)

b. Resultado del lanzamiento de dado
c. Movimiento realizado (incluyendo efectos de escaleras, serpientes o eventos)
d. Turno actual (qué jugador debe jugar)
e. Cartas disponibles para cada jugador
f. Monedas disponibles para cada jugador
g. Notificaciones de eventos especiales
h. Resultado de uso de cartas especiales
i. Actualizaciones de tiempo restante

4. **Sincronización periódica**
    a. Heartbeat para verificar conexión
    b. Estado completo del juego para jugadores que se reconectan

12. Diagrama Entidad-Relación (ER)

I. Entidades

1. Usuario

a. UsuarioID (PK): UUID
b. NombreUsuario: VARCHAR(50) UNIQUE
c. Email: VARCHAR(100) UNIQUE
d. Contraseña: VARCHAR(255) [Hash de contraseña]
e. Avatar: VARCHAR(255) [URL o referencia]
f. FechaRegistro: TIMESTAMP
g. UltimaConexion: TIMESTAMP
h. Estado: ENUM('activo', 'suspendido', 'inactivo')
i. Rol: ENUM('jugador', 'administrador', 'moderador')

2. Partida

a. PartidaID (PK): UUID
b. Nombre: VARCHAR(100)
c. Tipo: ENUM('pública', 'privada')
d. CodigoInvitacion: VARCHAR(10) [Sólo para privadas]

e. EstadoPartida: ENUM('en_espera', 'en_curso', 'finalizada')

f. MaxJugadores: INTEGER

g. FechaCreacion: TIMESTAMP

h. FechaInicio: TIMESTAMP [Null hasta que comience]

i. FechaFin: TIMESTAMP [Null hasta que termine]

j. TiempoEspera: INTEGER [En segundos]

k. PropietarioID (FK): UUID [Referencia a Usuario]

3. JugadorPartida

a. JugadorPartidaID (PK): UUID

b. UsuarioID (FK): UUID [Referencia a Usuario]

c. PartidaID (FK): UUID [Referencia a Partida]

d. PosicionActual: INTEGER [1-100]

e. Monedas: INTEGER

f. EnTurno: BOOLEAN

g. Orden: INTEGER [Orden de juego]

h. Estado: ENUM('activo', 'abandonó')

i. UltimoTurno: TIMESTAMP

4. Carta

a. CartaID (PK): UUID

b. Tipo: ENUM('doble_dado', 'intercambiar_posicion', 'saltar_serpiente', 'retroceder_turno')

c. Descripcion: TEXT

d. CostoMonedas: INTEGER

e. IconoURL: VARCHAR(255)

5. CartaJugador

a. CartaJugadorID (PK): UUID

b. JugadorPartidaID (FK): UUID [Referencia a JugadorPartida]

c. CartaID (FK): UUID [Referencia a Carta]

d. ObtenidasPor: ENUM('compra', 'gratuita', 'evento')

e. FechaObtencion: TIMESTAMP

f. FechaUso: TIMESTAMP [Null si no se ha usado]

g. Activa: BOOLEAN

6. **Tablero**

    a. TableroID (PK): UUID

    b. PartidaID (FK): UUID [Referencia a Partida]

    c. Configuracion: JSONB [Almacena escaleras, serpientes, casillas especiales]

7. **CasillaEspecial**

    a. CasillaEspecialID (PK): UUID

    b. TableroID (FK): UUID [Referencia a Tablero]

    c. Posicion: INTEGER [1-100]

    d. Tipo: ENUM('ganar_monedas', 'perder_monedas', 'carta_gratis', 'perder_turno', 'robar_carta')

    e. Valor: INTEGER [Cantidad de monedas o cartas afectadas]

    f. Descripcion: TEXT

8. **MovimientoJugador**

    a. MovimientoID (PK): UUID

    b. JugadorPartidaID (FK): UUID [Referencia a JugadorPartida]

    c. TipoMovimiento: ENUM('dado', 'escalera', 'serpiente', 'carta', 'evento_especial')

    d. PosicionInicial: INTEGER

    e. PosicionFinal: INTEGER

    f. ValorDado: INTEGER [Null si no es lanzamiento de dado]

    g. CartaUsadaID (FK): UUID [Referencia a CartaJugador, Null si no es tipo carta]

    h. FechaMovimiento: TIMESTAMP

    i. TurnoNumero: INTEGER

9. **TiendaPartida** *Sujeto a cambio. Puede ser que al comprar una carta la elección de ésta sea aleatoria.

    a. TiendaID (PK): UUID

    b. PartidaID (FK): UUID [Referencia a Partida]

    c. CartaID (FK): UUID [Referencia a Carta]

d. DisponibleDesde: TIMESTAMP

e. DisponibleHasta: TIMESTAMP [Null si siempre disponible]

10. Estadistica

a. EstadisticaID (PK): UUID

b. UsuarioID (FK): UUID [Referencia a Usuario]

c. PartidasJugadas: INTEGER

d. PartidasGanadas: INTEGER

e. PartidasAbandonadas: INTEGER

f. MayorPosicion: INTEGER

g. TotalMonedas: INTEGER

h. TotalCartasUsadas: INTEGER

i. TurnosJugados: INTEGER

j. UltimaActualizacion: TIMESTAMP

II. Relaciones

1. **Usuario-Partida (1-N):** Un usuario puede crear múltiples partidas, pero cada partida tiene un único propietario.
2. **Usuario-JugadorPartida (1-N):** Un usuario puede participar en múltiples partidas como jugador.
3. **Partida-JugadorPartida (1-N):** Una partida puede tener múltiples jugadores (2-4 según configuración).
4. **Partida-Tablero (1-1):** Cada partida tiene un tablero único con su configuración.
5. **Tablero-CasillaEspecial (1-N):** Un tablero tiene múltiples casillas especiales.
6. **JugadorPartida-CartaJugador (1-N):** Un jugador en una partida puede tener múltiples cartas.
7. **Carta-CartaJugador (1-N):** Un tipo de carta puede estar asociado a múltiples jugadores.
8. **JugadorPartida-MovimientoJugador (1-N):** Un jugador puede realizar múltiples movimientos en una partida.
9. **Partida-TiendaPartida (1-N):** Una partida tiene múltiples cartas disponibles en su tienda. *Sujeto a cambio. Puede ser que al comprar una carta la elección de ésta sea aleatoria.

10. **Carta-TiendaPartida (1-N):** Una carta puede estar disponible en la tienda de múltiples partidas. *****Sujeto a cambio. Puede ser que al comprar una carta la elección de ésta sea aleatoria.
11. **Usuario-Estadistica (1-1):** Cada usuario tiene un registro de estadísticas asociado.

III. Mapeo a PostgresSQL

Se adjunta el archivo *mapeo_postgresql.sql* con el código SQL que representa el mapeo del diagrama ER. Además, se presenta a continuación un detalle de dicho mapeo explicado en palabras:

1. **Tipos ENUM personalizados**: Para garantizar la consistencia de datos en campos como estados, tipos de cartas, y roles.
2. **Claves primarias UUID**: Uso de identificadores universales para mayor seguridad y evitar problemas de secuencia.
3. **Restricciones de integridad**:
    a. Claves foráneas con cascada para mantener integridad referencial
    b. Restricciones CHECK para validar rangos de valores (ej: posiciones en el tablero entre 1-100)
    c. Restricciones UNIQUE para evitar duplicados
4. **Valores por defecto**: Para facilitar la creación de registros con valores iniciales coherentes.
5. **Índices**: Creados en columnas frecuentemente consultadas para mejorar el rendimiento.
6. **Campo JSONB**: Para almacenar configuraciones complejas del tablero (ubicaciones de escaleras, serpientes).
7. **Triggers automáticos**:
    a. Actualización de última conexión cuando un usuario inicia una partida
    b. Actualización automática de estadísticas cuando finaliza una partida
8. **Relaciones** implementadas mediante claves foráneas que reflejan el diagrama ER.
9. **Auditoría temporal**: Campos de fecha y hora para registrar creación, modificación y finalización de eventos.

13. Diagrama de clases

El diagrama de clases presentado para el juego, muestra la estructura completa del sistema, incluyendo las clases principales del dominio, los servicios gestores y los controladores para la interfaz web. Se adjunta el archivo *Diagrama de clase - entrega 1.svg* con el diagrama. A continuación, se explica detalladamente cada componente:

I. Clases del dominio

1. **Usuario**: Representa a los usuarios registrados en el sistema. Almacena información como credenciales, avatar, y estado. Incluye métodos para gestionar la cuenta y participar en partidas.
2. **Partida**: Encapsula una sesión de juego, con información sobre su estado, participantes y configuración. Gestiona el flujo del juego y las reglas generales.
3. **Tablero**: Representa el tablero de 10x10 casillas. Almacena la configuración de escaleras, serpientes y casillas especiales, y maneja la lógica de movimiento.
4. **Jugador**: Representa la participación de un usuario en una partida específica. Almacena su posición, recursos (monedas) y estado actual, además de gestionar sus acciones de juego.
5. **Casilla**: Representa cada posición en el tablero. Define comportamientos según su tipo (normal, escalera, serpiente o especial).
6. **CasillaEspecial**: Extiende la clase Casilla, añadiendo funcionalidades específicas para casillas con eventos especiales como ganar monedas o perder turnos.
7. **Carta**: Define los tipos de cartas especiales y sus efectos, como tirar doble dado o intercambiar posición.
8. **CartaJugador**: Representa la posesión de una carta por un jugador específico, controlando su estado (activa/usada).
9. **Movimiento**: Registra cada acción de movimiento realizada por los jugadores para mantener un historial de la partida.
10. **Dado**: Encapsula la lógica del lanzamiento de dados con su generador de números aleatorios.
11. **TiendaPartida**: Gestiona el catálogo de cartas disponibles para compra en una partida específica. *Sujeto a cambio. Puede ser que al comprar una carta la elección de ésta sea aleatoria.
12. **Estadistica**: Almacena métricas de desempeño de cada usuario a lo largo del tiempo.

II. Clases de servicio

1. **GestorPartidas**: Coordina la creación, inicio y finalización de partidas, así como la gestión de jugadores.
2. **GestorTableros**: Maneja la creación y configuración de tableros, incluyendo la colocación de escaleras, serpientes y casillas especiales.
3. **GestorCartas**: Administra la creación, asignación y uso de cartas especiales en el juego.
4. **GestorUsuarios**: Gestiona el registro, autenticación y administración de usuarios del sistema.

III. Controladores

1. **ControladorPartidas**: Maneja las solicitudes web relacionadas con la creación y gestión de partidas.
2. **ControladorJuego**: Procesa las interacciones del jugador durante una partida activa.
3. **ControladorUsuarios**: Gestiona el registro, inicio de sesión y perfil de usuarios.

IV. Relaciones principales

- **Herencia**: CasillaEspecial hereda de Casilla, extendiendo su funcionalidad.
- **Composición**: Una Partida está compuesta por un Tablero y múltiples Jugadores.
- **Asociación**: Un Usuario participa en múltiples Partidas a través de la clase Jugador.
- **Uso**: Los Gestores utilizan las clases del dominio para implementar la lógica de negocio.
- **Dependencia**: Los Controladores dependen de los Gestores para procesar las solicitudes web.

V. Patrones de diseño implementados:

1. **Patrón MVC (Modelo-Vista-Controlador)**: Separación entre las clases del dominio (Modelo), los controladores y las vistas (implícitas en los métodos de los controladores).
2. **Patrón Singleton** (implícito): Los gestores probablemente serán implementados como singletons para mantener estado global.
3. **Patrón Strategy**: La variabilidad en los efectos de las cartas y casillas especiales sugiere una implementación basada en estrategias.
4. **Patrón Observer**: Para notificar cambios en el estado del juego a todos los jugadores conectados.
5. **Patrón Factory**: Implícito en la creación de diferentes tipos de cartas y casillas especiales.

14. Tablero Kanban (Github projects)

Se distribuyeron las tareas y se completaron utilizando el tablero de GitHub Projects. El link es: https://github.com/IIC2513/Los3_front_s2/projects?query=is%3Aopen