TEC | Tecnológico de Costa Rica

Instituto Tecnológico de Costa Rica

Lenguajes de Programación

Proyecto 4: Juego LuiKi Kart

Profesor:

Allan Rodríguez Dávila

Estudiantes:

Jimena Méndez Morales - 2023113347

Ricardo Arce Aguilar - 2023215990

I Semestre, 2025

Tabla de Contenidos

Tabla de Figuras

Pruebas de funcionalidad

1. Ingreso del nickname: El usuario ingresa el nombre que desea para el juego.

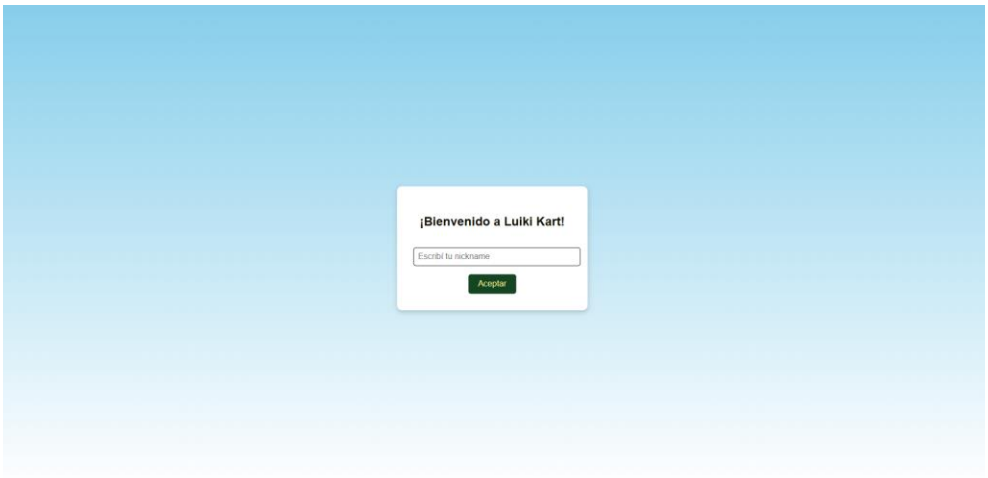


Ilustración 1. Ingreso de usuario

2. Menú principal: El usuario puede acceder a distintas secciones:

    - Crear una partida
    - Unirse a una partida
    - Ver estadísticas de la partida que jugó
    - Ver ranking

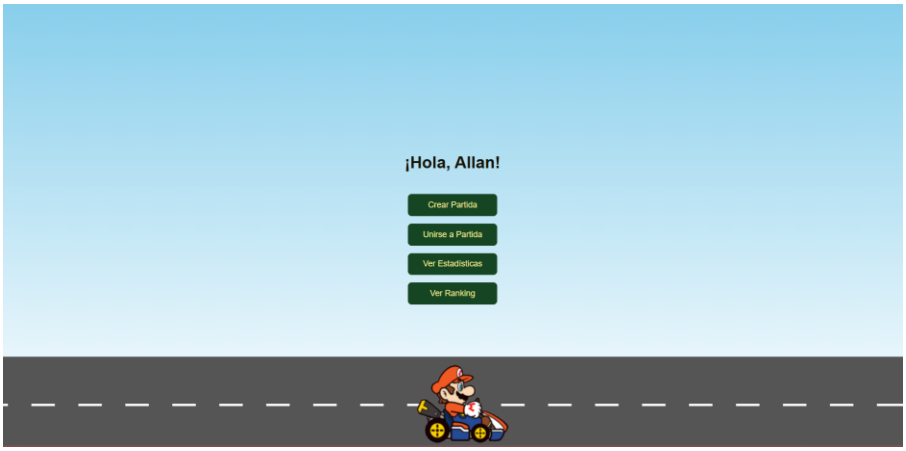


Ilustración 2. Menú principal

3. Crear partida: Al crear una partida, se ingresan los campos como qué tipo de juego desea jugar, la pista, la cantidad de rondas o vueltas y la cantidad de jugadores.

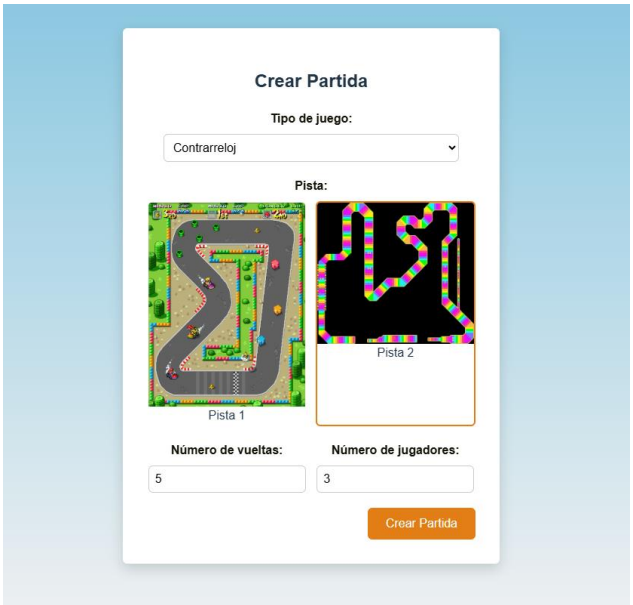


Ilustración 3. Creación de partida

4. Unirse a una partida: Una vez creada esa partida, se pueden crear otros usuarios para unirse a ella como el ejemplo.

Ilustración 4. Verificar estado del juego

5. Lobby: Al momento que se une un usuario o el usuario crea la partida, entonces se puede visualizar el lobby donde todos están a la espera, también es importante identificar que si no está la cantidad completa de número de jugadores, entonces no deja que los demás

estén listos como en la imagen 7 y si todos están listos, se notifica igualmente como la imagen 8.

Ilustración 5. Lobby

Ilustración 6. Error de jugadores preparados

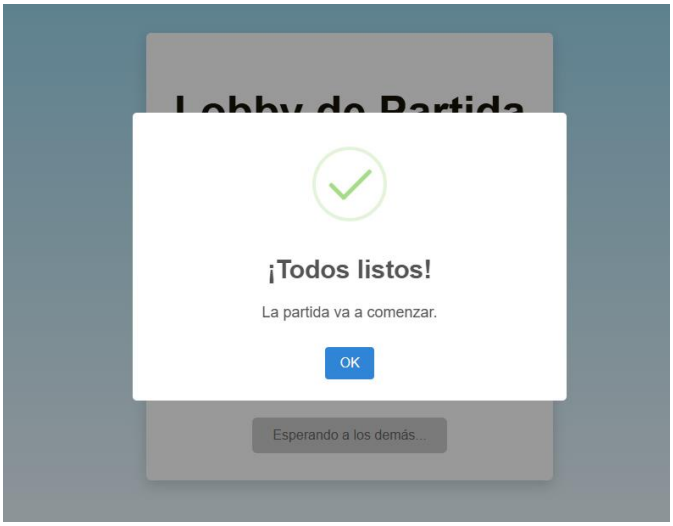


Ilustración 7. Todos los jugadores preparados

Descripción del problema

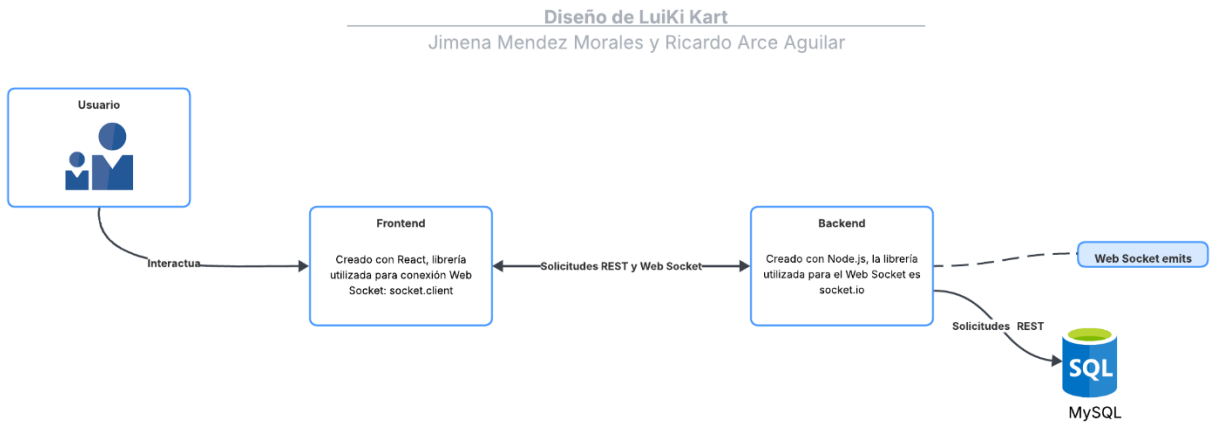
Se solicita recrear la funcionalidad de un juego de carreras multijugador. El sistema debe conformarse por 2 subsistemas: un backend, donde se ejecute toda la lógica del juego, y el frontend, donde cada jugador pueda elegir un nombre y crear o unirse a una partida. Se requiere también que se manejen múltiples partidas en simultaneo con varios jugadores mediante uso del protocolo WebSocket. Además, se solicitó que el frontend sea accesible desde una dirección ip pública mediante el uso de ngrok, cloudflare tunnel, tailscale, o similar.

Al crear una partida, el jugador debe poder elegir la cantidad de jugadores, la cantidad de vueltas, el mínimo de jugadores, la pista a recorrer, y el tipo de juego, que se refiere a si se ganará al completar cierta cantidad de vueltas o si será una partida cronometrada. Los vehículos se deben asignar aleatoriamente. Los nombres de los jugadores deben ser visibles durante la partida.

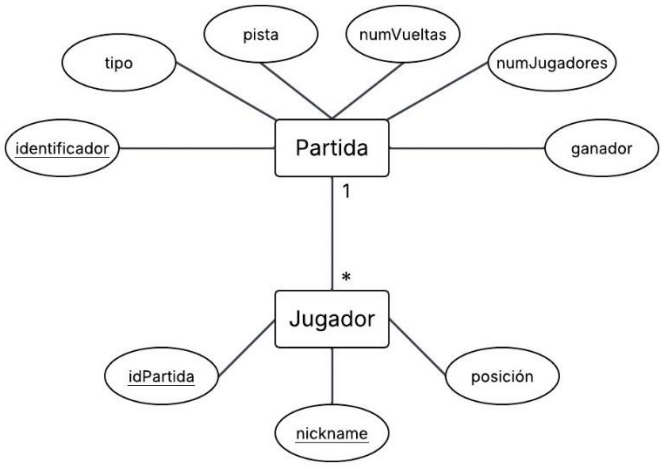
El jugador debe poder ver una sección de ranking de la partida, así como las estadísticas de todas las partidas jugadas anteriormente. Se debe mostrar, por cada partida, el nombre de cada jugador, su tiempo o cantidad de vueltas, su posición y el ganador de la partida.

Diseño del programa

- Arquitectura planeada del proyecto:



- Diagrama Entidad-Relación de la base de datos:



Librerías usadas

Backend (Node.js + Express)

- **mysql2/promise**: Esta librería permite conectarse a bases de datos MySQL utilizando Promesas. Facilita el uso de async/await, lo que hace que las consultas a la base de datos sean más limpias y fáciles de manejar.
- **dotenv**: Se encarga de cargar variables de entorno desde un archivo .env. Esto es útil para proteger información sensible como contraseñas, claves API o configuraciones del entorno.
- **express**: Framework web ligero y rápido para construir APIs REST. Simplifica la gestión de rutas, solicitudes HTTP y middlewares.

- **http**: Módulo nativo de Node.js que permite crear un servidor HTTP. Se usa para integrarlo con socket.io y permitir la comunicación en tiempo real.
- **socket.io**: Biblioteca que permite implementar WebSockets para comunicación bidireccional entre el cliente y el servidor. Se utiliza para sincronizar eventos del juego como el estado de los jugadores.
- **cors**: Middleware que permite manejar políticas de acceso entre diferentes dominios (CORS). Es esencial cuando el frontend y el backend están en servidores diferentes.
- **router (./routes.js)**: Archivo que contiene las rutas REST definidas para interactuar con las distintas funcionalidades del sistema (crear partidas, asignar jugadores, etc.).

Frontend (React.js)

- **react**: Biblioteca principal para construir interfaces de usuario en componentes reutilizables. Se encarga del renderizado y la gestión del estado visual de la aplicación.
- **react-router-dom**: Permite implementar navegación entre diferentes vistas dentro de una aplicación de una sola página (SPA). Se utiliza para manejar rutas como /lobby, /home, /game, etc.
- **sweetalert2**: Biblioteca para mostrar alertas modernas, visualmente atractivas y personalizables, como notificaciones de error o confirmaciones.
- **createContext**: Parte del sistema de contexto de React. Se usa para compartir información global (como el socket) entre componentes sin pasar props manualmente.
- **BrowserRouter**:

- Componente que envuelve la aplicación y permite el manejo del historial del navegador y las rutas amigables.

Análisis de resultados

Objetivos	**Alcanzado / no alcanzado**
El jugador debe poder elegir un nickname.	
El jugador debe poder crear una partida.	
El jugador debe poder seleccionar el tipo de juego de una partida, ya sea en modo versus o modo contrarreloj.	
El jugador debe poder seleccionar el número de vueltas a recorrer en una partida.	
El jugador debe poder seleccionar el número de jugadores en una partida.	
El jugador debe poder unirse a una partida existente.	
El jugador debe poder marcarse como “preparado”.	

El jugador debe poder cambiar la dirección de su vehículo usando los botones de flechas.	No se logró por temas de tiempo
El vehículo del jugador debe detenerse si choca con una pared o el vehículo de otro jugador.	No se logró por temas de tiempo
El jugador debe poder ver el nombre de los demás jugadores durante la partida.	No se logró por temas de tiempo
El jugador debe poder ver una advertencia cuando se esté desplazando en sentido antihorario.	No se logró por temas de tiempo
El jugador debe poder ver el ganador de la partida.	No se logró por temas de tiempo
El jugador debe poder ver una sección de estadísticas en la que se detalle, por cada juego, el nombre de los jugadores, su tiempo o posición, la pista y el identificador de partida.	No se logró por temas de tiempo
El jugador debe poder ver una sección de ranking en la que se detalle, por cada partida, el ganador, su tiempo, la pista, el número de vueltas y el identificador de partida.	No se logró por temas de tiempo

Bitácora

Enlace del repositorio: https://github.com/jimendezm/proyecto-4-lp.git