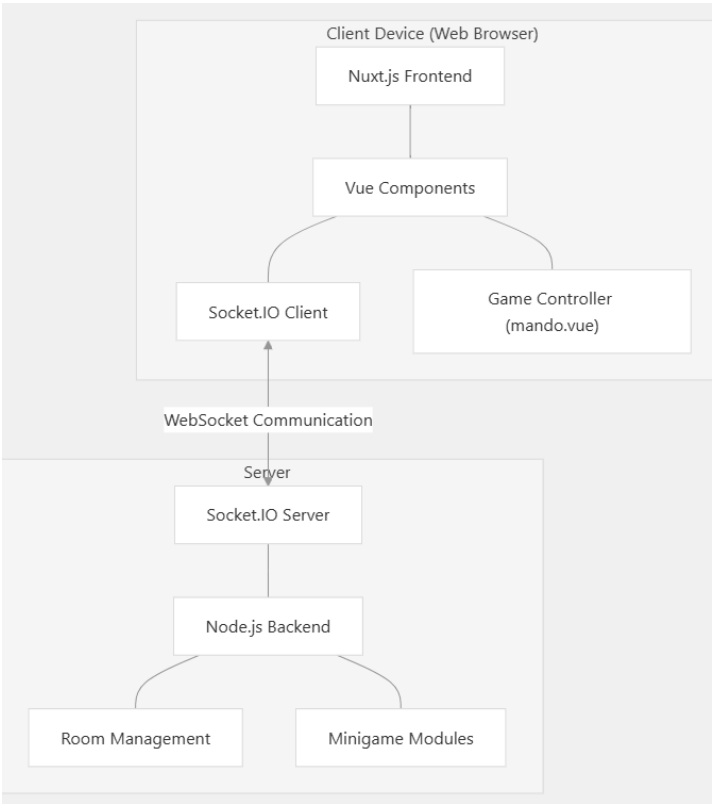
Disseny DuelParty

Elihú Valdelomar
Diego Mujica

Índex

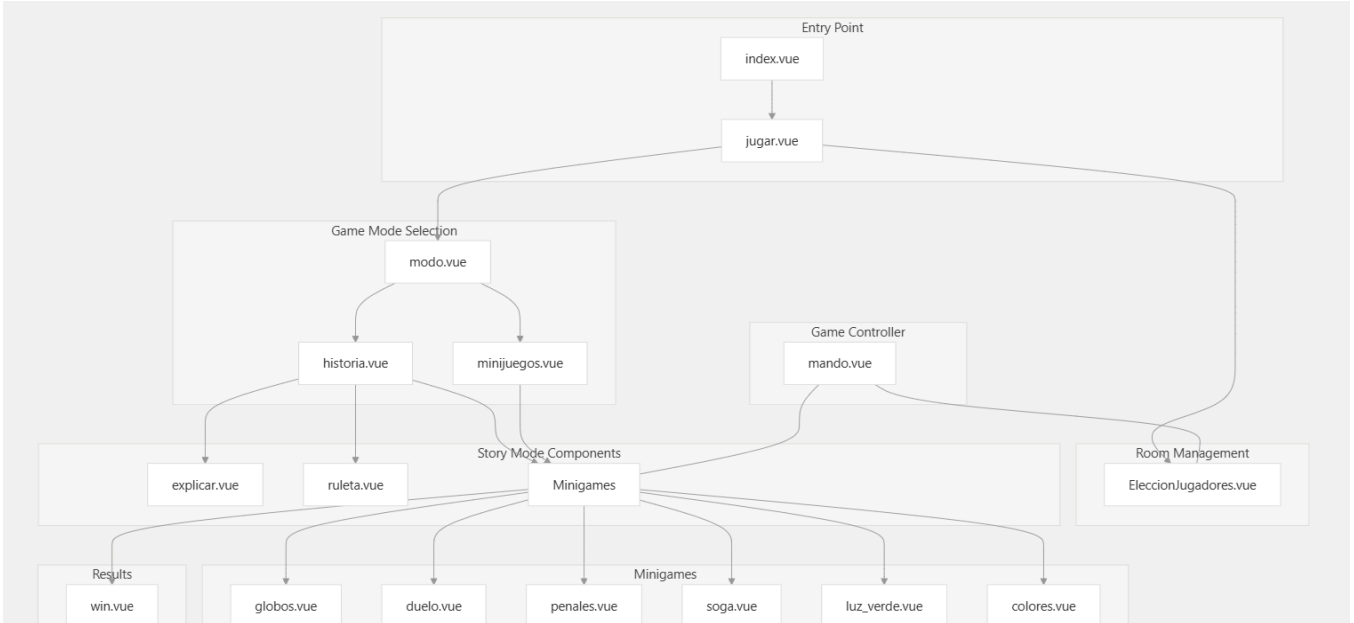
Arquitectura Basica

DuelParty segueix una arquitectura client-servidor amb comunicació en temps real a través de WebSockets. El frontend s'executa als navegadors web dels jugadors (tant a la pantalla de joc compartida com als dispositius mòbils), mentre que el backend gestiona l'estat del joc, la lògica de la sala i sincronitza les accions dels jugadors entre dispositius.



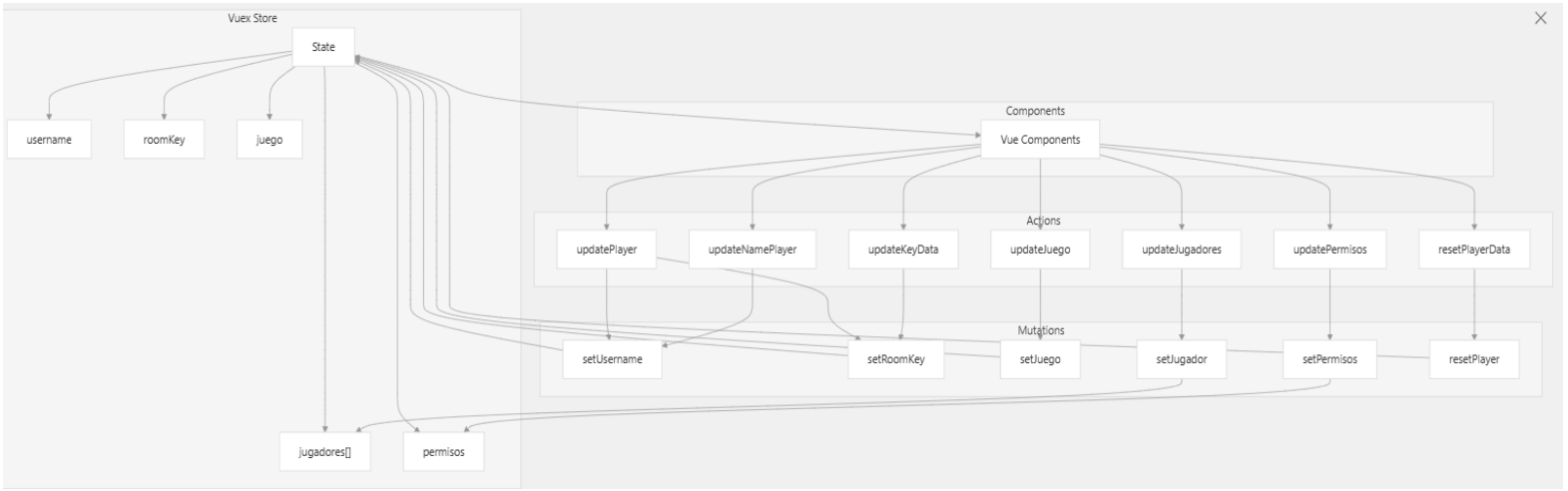
Arquitectura del Frontend

El frontend de DuelParty segueix una estructura de components jeràrquica que facilita el flux del joc des de la pàgina de destinació fins a la jugabilitat.



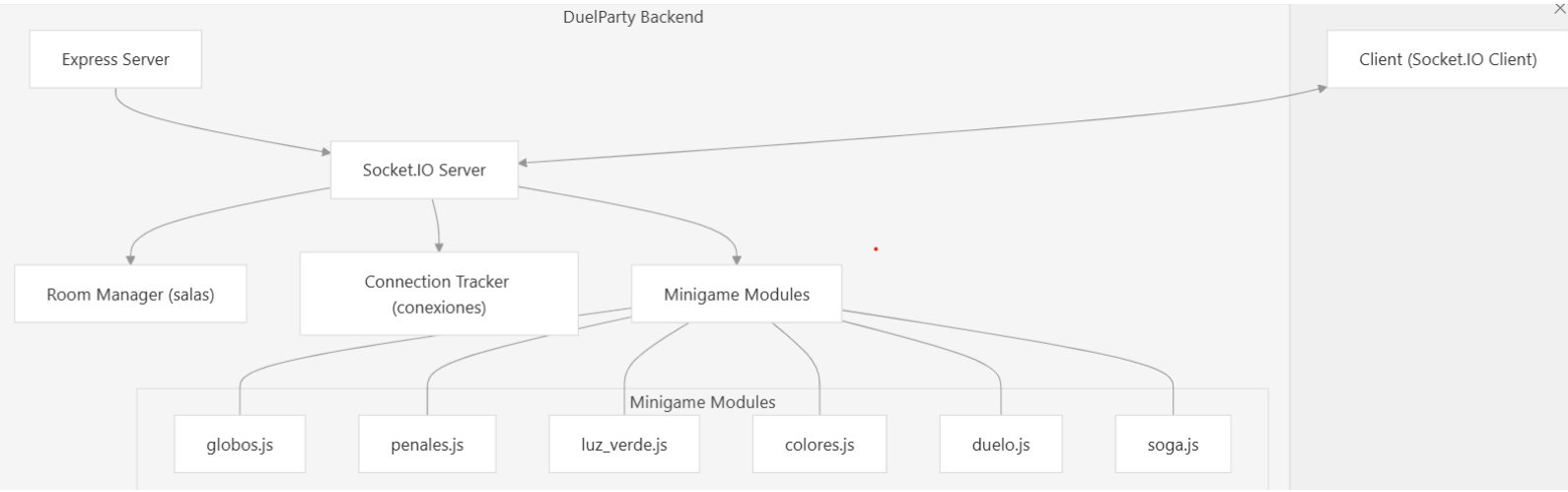
Gestió d'estat

La gestió de l'estat en DuelParty s'implementa amb Vuex. Manté l'estat global de l'aplicació, inclosa la informació de l'usuari, els detalls de la sala i l'estat del joc.

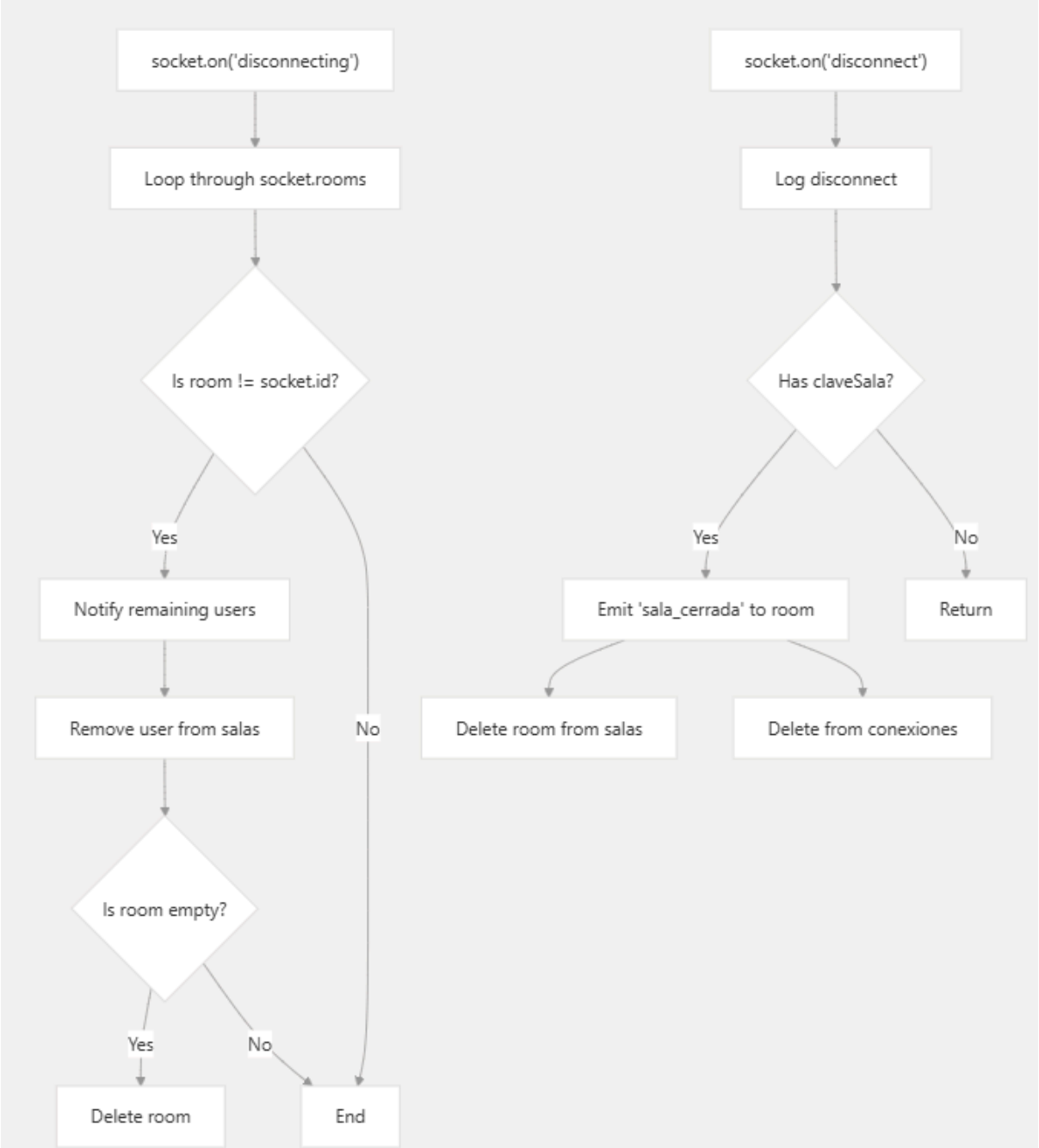


Arquitectura del Backend

El backend de DuelParty està construït sobre Node.js amb Express i Socket.IO per proporcionar funcionalitat de joc multijugador en temps real. Gestiona sales de joc, gestiona les connexions dels clients i coordina l'intercanvi de dades entre jugadors durant els minijocs.

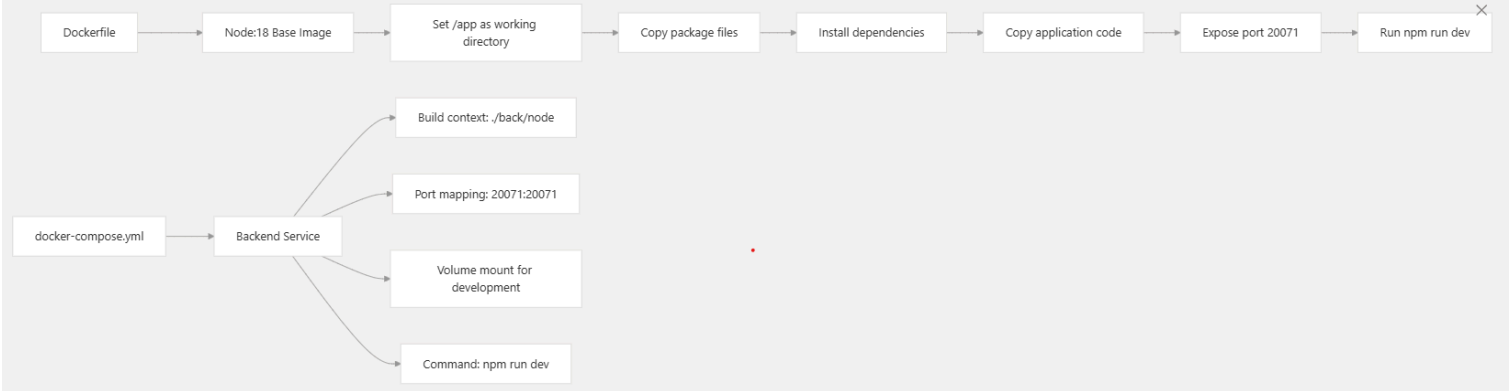


Gestió de connexions i desconnexions

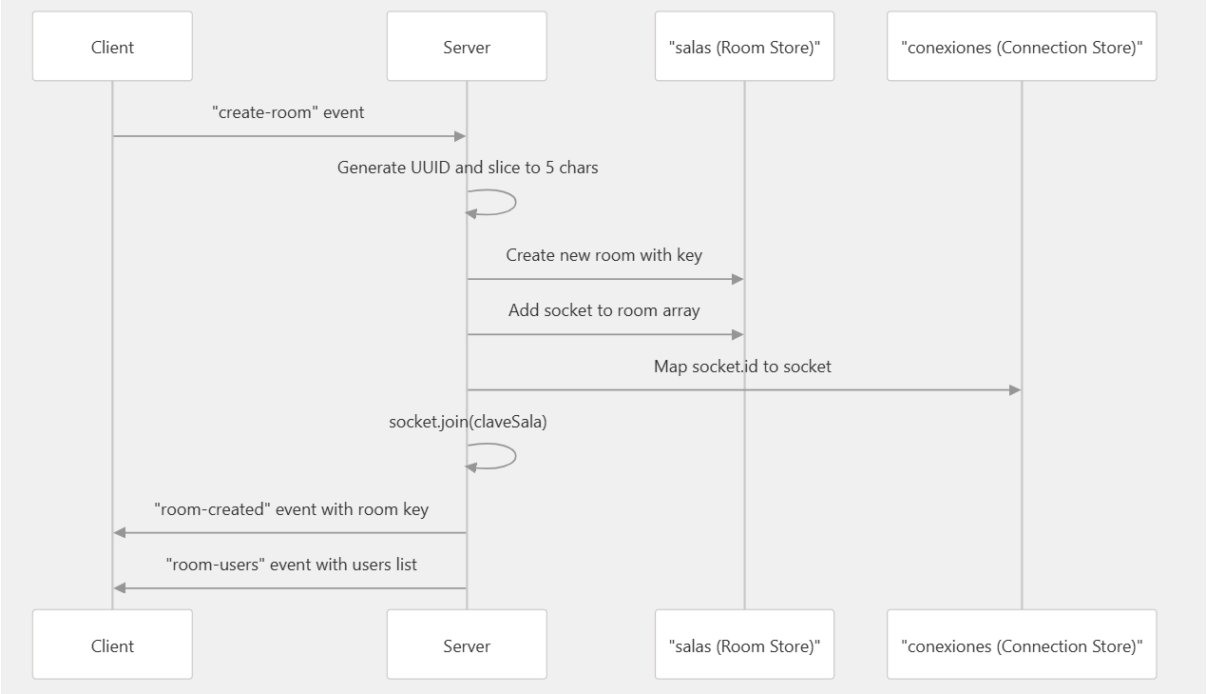


Configuració Docker

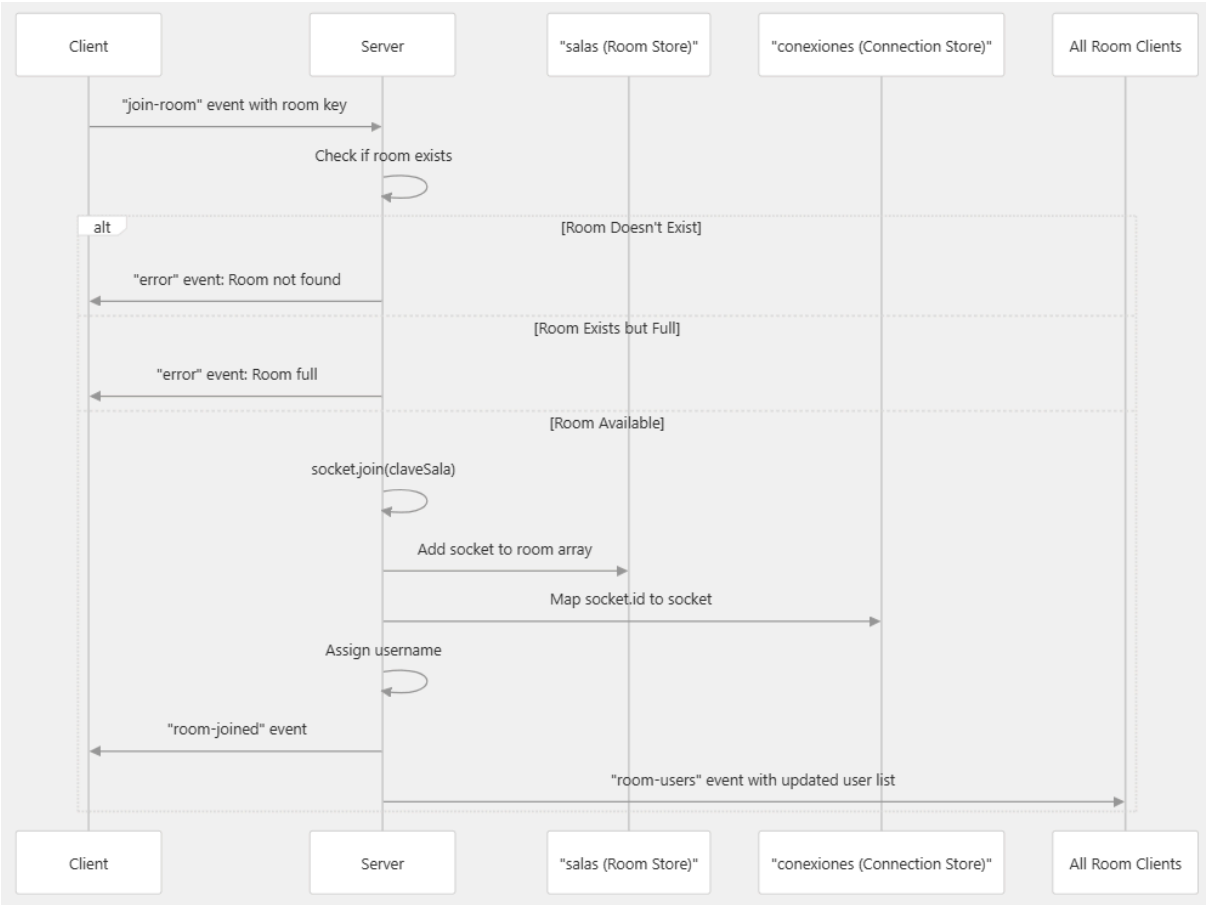
El servei de backend està configurat a docker-compose.yml per permetre la recàrrega en calent durant el desenvolupament muntant el directori local i utilitzant nodemon.



Flux de creació de salas



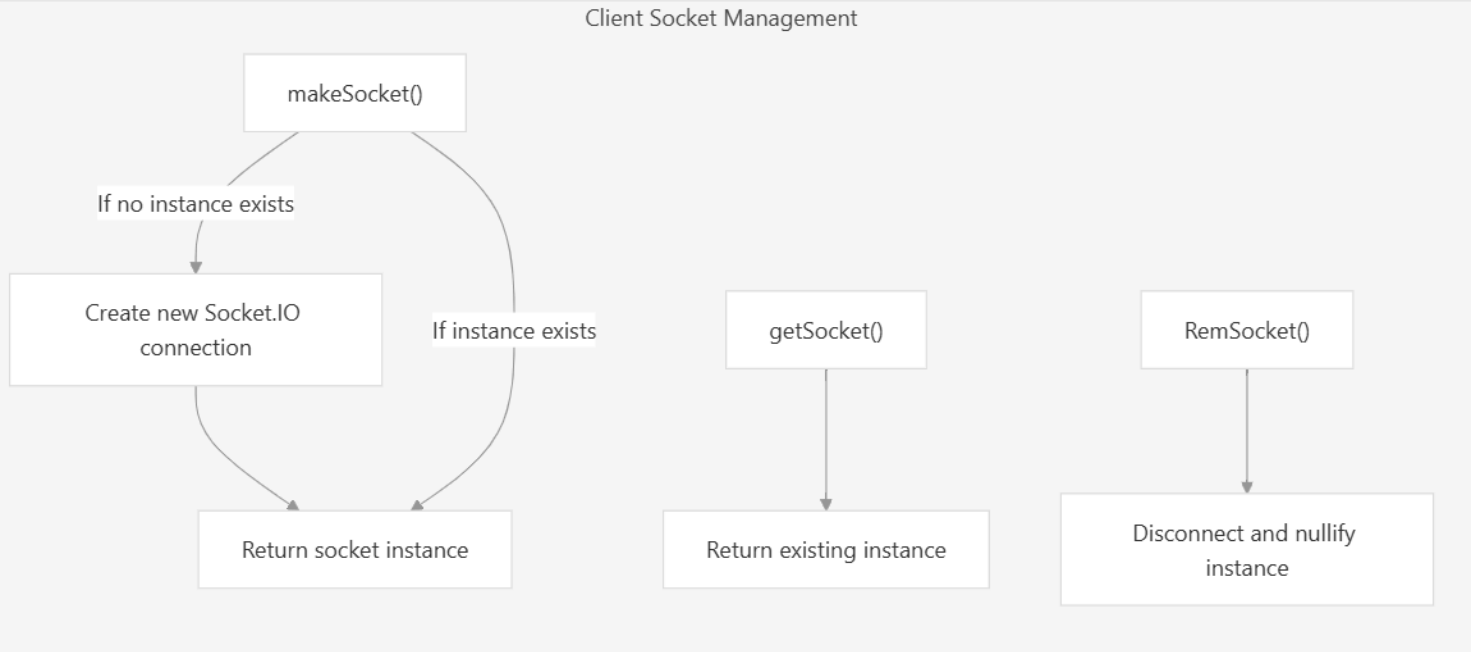
Flux d’unió de salas



Comunicació Socket.io

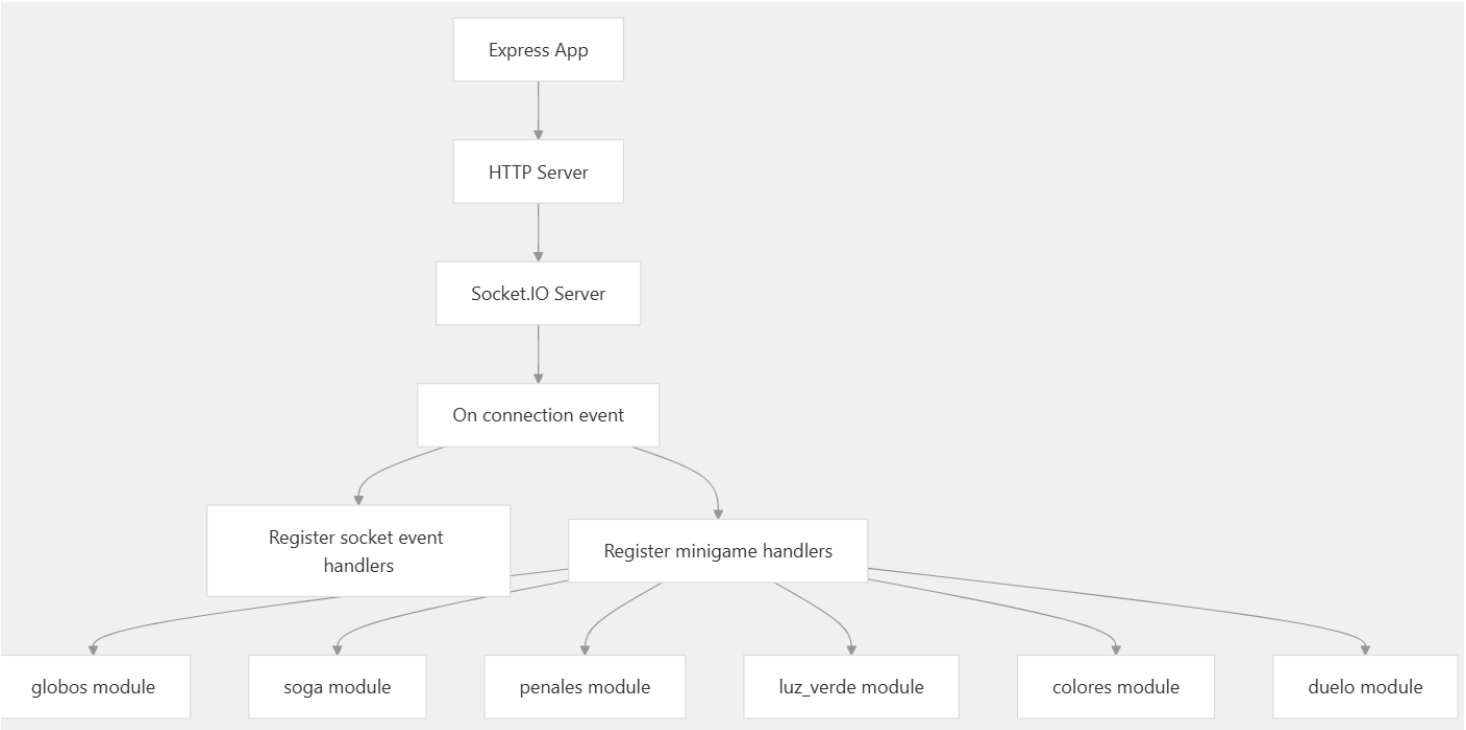
Configuració del client

El frontend estableix connexions de socket mitjançant un patró implementat a socket.js. Aquest patró garanteix que només existeix una connexió de socket activa per client.

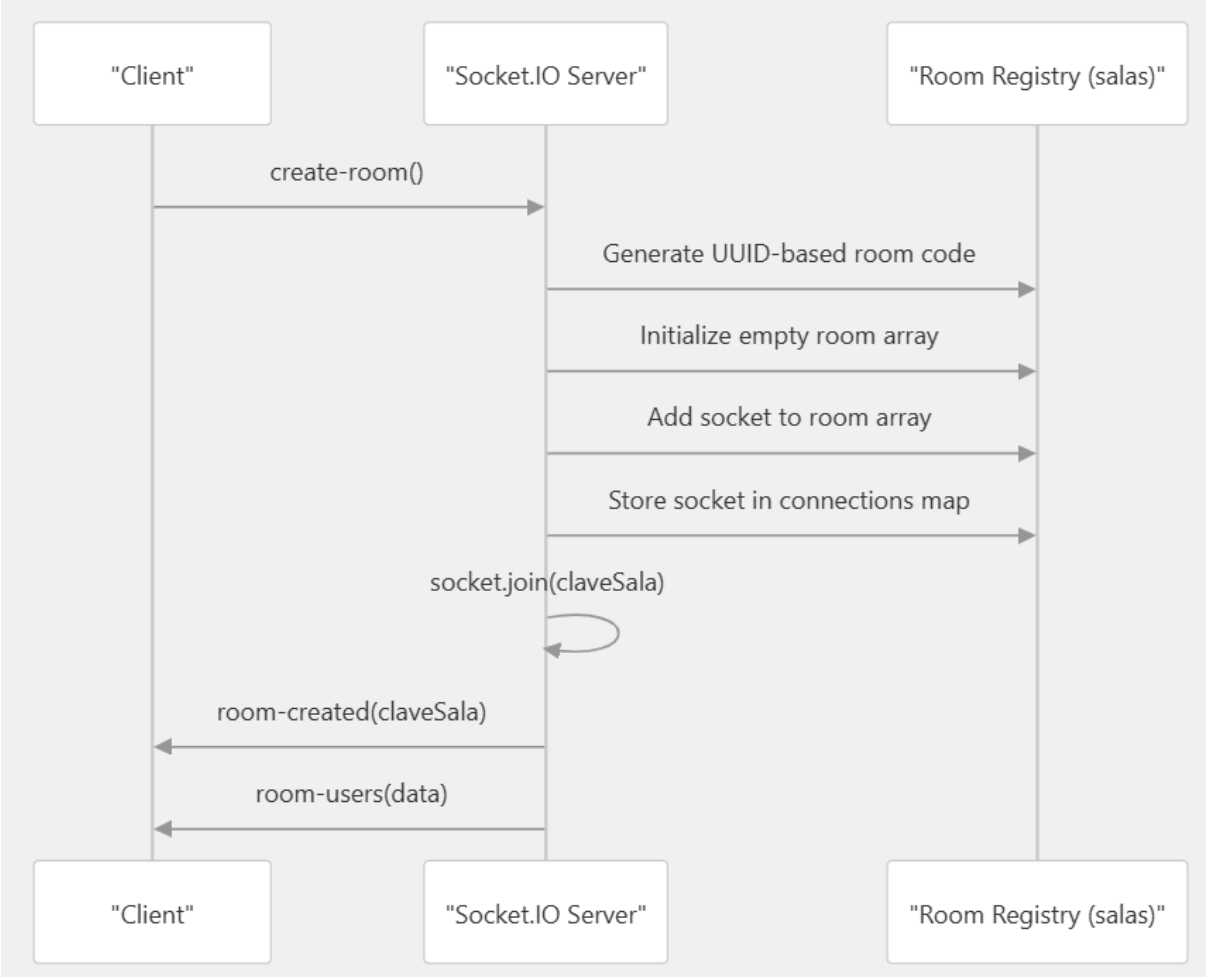


Configuració del servidor

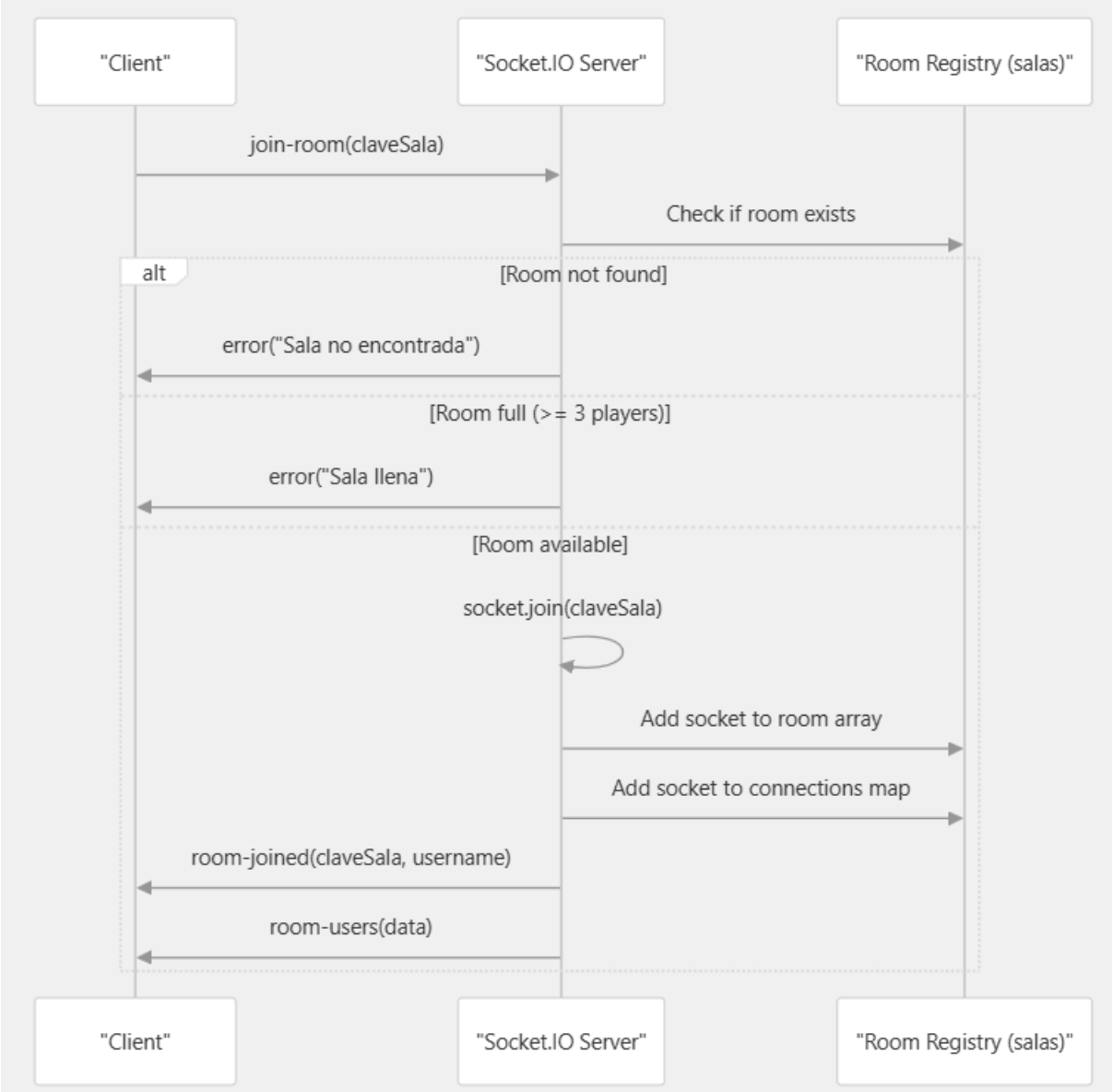
El servidor crea una instància Socket.IO connectada a un servidor HTTP amb configuració CORS per acceptar connexions de qualsevol origen.



Procés de creació de salas

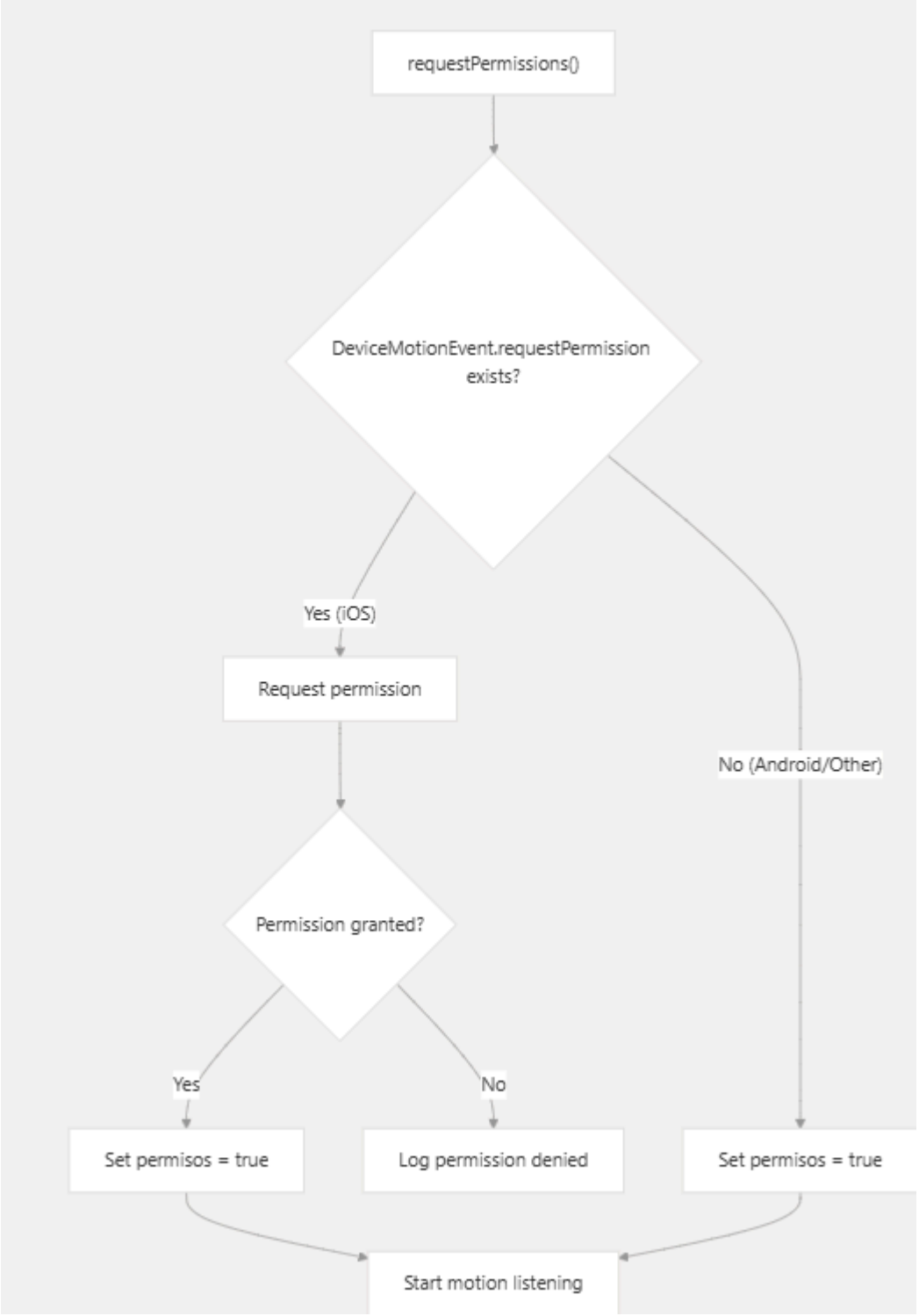


Procés d'unió de salas



Gestió de permisos per a dispositius iOS

El sistema gestiona els requisits de permisos especials per accedir al moviment del dispositiu en dispositius iOS:



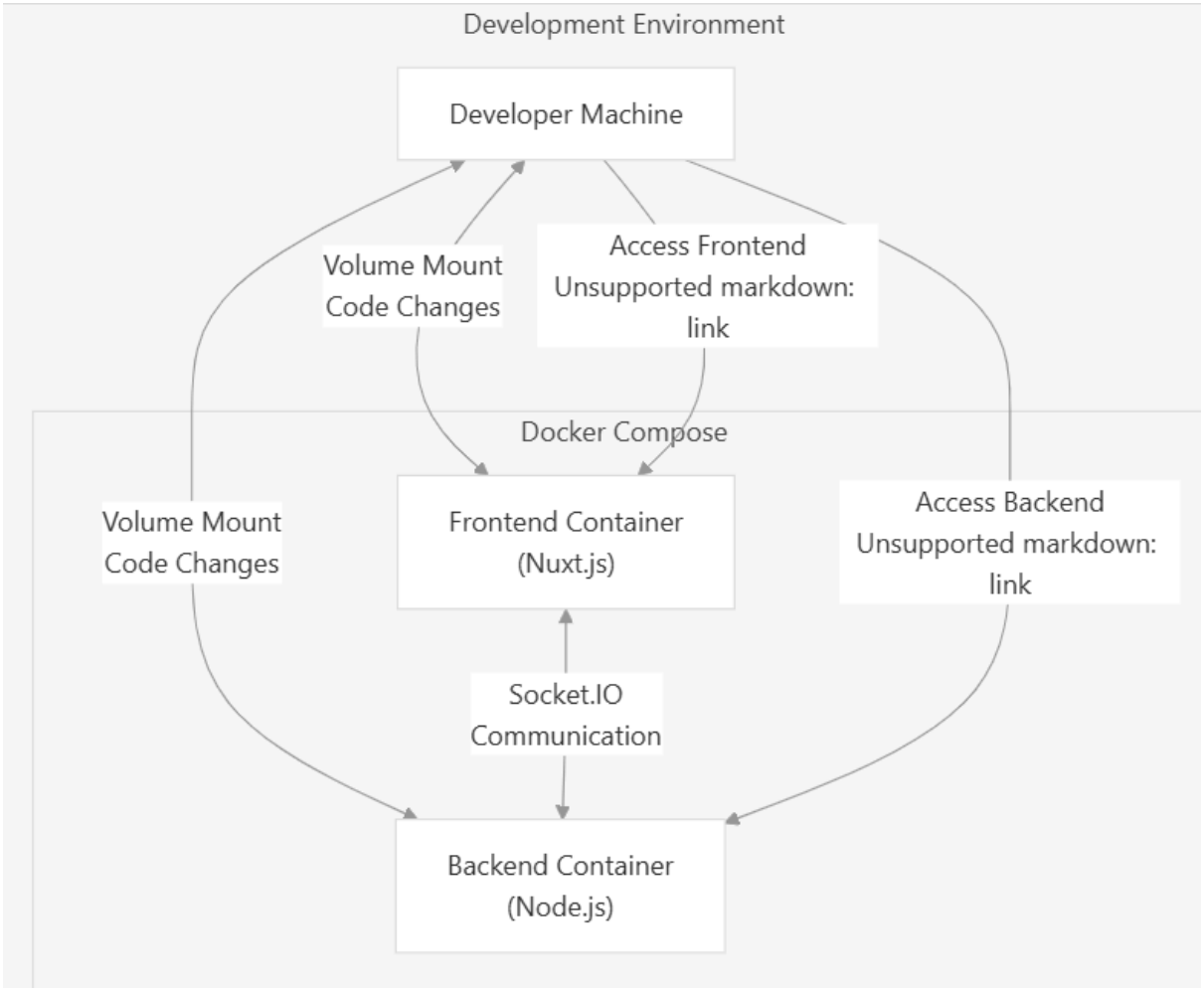
Events de gestió de sales

Nom Event	Direcció	Propòsit
`create-room`	Client → Server	Request per crear una nova sala
`room-created`	Server → Client	Confirmació amb la room key
`join-room`	Client → Server	Request pero poder entrar en una sala existent
`room-joined`	Server → Client	Confirmació de que s'ha pogut entrar sala
`room-users`	Server → Clients	Actualiza a tots els participants actuals de la sala

Events de sockets

Nom del event	Direcció	Propòsit
`move`	Client ↔ Server ↔ Clients	Moviment/acció genèrics
`pagina`	Client → Server → Clients	Canvia la pàgina/vista per a tots els jugadors
`minijuego`	Client → Server → Clients	Selecciona o canvia el minijoc actual
`enviar_duelo`, `enviar_soga`, etc.	Client → Server	Esdeveniments de control específics del joc

Configuració de Docker



El fitxer docker-compose.yml defineix els contenidors, les assignacions de ports, els volums i les ordres d'inici:

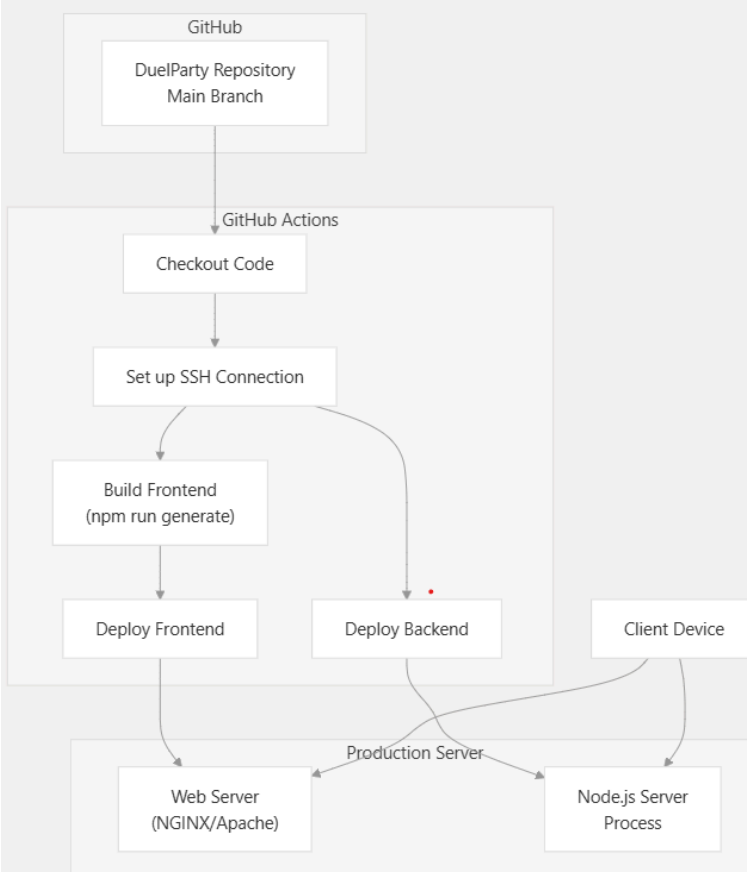
- Frontend (Nuxt.js):
  Port: 3000
  Volum: Assigna el directori local ./front a /app al contenidor
  Comanda: npm run dev

- Backend (Node.js):
  Port: 20071
  Volum: Assigna el directori local ./back/node a /app al contenidor
  Comanda: npm run dev.

Les assignacions de volums garanteixen que els canvis als fitxers de codi locals es reflecteixin immediatament als contenidors, permetent la recàrrega en calent durant el desenvolupament.

Implementació de producció

El projecte utilitza GitHub actions per a la implementació automatitzada a producció quan els canvis es publiquen a la branca principal.



Flux de treball d'accions de GitHub

El procés de desplegament s'automatitza mitjançant un flux de treball d'accions de GitHub definit a .github/workflows/deploy.yml. El flux de treball inclou:

1. Disparador: S'activa en enviar a la branca principal
2. Entorn: Utilitza la darrera versió d'Ubuntu com a executable
3. Passos:
    - Codi de descàrrega
    - Configura l'accés SSH al servidor de producció
    - Desplega el backend (Node.js)
    - Crea i desplega el frontend (Nuxt.js)
    - El flux de treball utilitza SSH i rsync per transferir fitxers al servidor de producció.

Minijocs

DuelParty inclou sis minijocs diferents en què els jugadors poden competir. Aquests minijocs són accessibles mitjançant la selecció directa al menú de minijocs o com a part del mode de

joc d'escalada de muntanya. Cada minijoc segueix un patró arquitectònic comú però implementa mecàniques de joc úniques.

Icon	Nom	Descripció	Component
🎈	Globos	Joc de fer esclatar globus	globos.vue
🔫	Duelo oeste	Joc de cronometratge de duel occidental	duelo.vue
⚽	Penales	Partit de tanda de penals	penales.vue
🫳	Tirar de la soga	Joc d'estira-i-la-corda	soga.vue
🚶	Luz verde, luz roja	Joc de llum vermella, llum verda	luz_verde.vue
🎨	Colores	Joc de coincidència de colors	colores.vue

Referències:

https://github.com/inspedralbes/prj-final-duel_party
https://deepwiki.com/inspedralbes/prj-final-duel_party/1-overview