ACTIVITAT
Objectius: ■ Processat de dades amb JSON en MongoDB
Instruccions: ■ Resoleu els exercicis proposats i pengeu el repositori a GitHub
Criteris d'avaluació: ■ Cada pregunta té el mateix pes ■ Es valorarà la presentació i els comentaris al codi
Entrega: ■ A Moodle enllaç a repositori GitHub. - A dins del repositori, en un directori anomenat "doc" inclout aquest document completat en format pdf i amb el nom "memoria.pdf"
Repositori d'exemple: ■ https://github.com/jpala4-ieti/DAM-M06-UF03-Punt-Partida-NodeJS

Exercicis

Exercici 1. Mini-Servidor WebSocket de Registre d'Esdeveniments de Joc amb Client Node.js (10 punts)

Crear un servidor WebSocket bàsic en Node.js que escolti connexions entrants. El servidor rebrà missatges en format JSON simulant el moviment 2D d'un jugador en una partida des d'un client Node.js i els emmagatzemarà en una col·lecció de MongoDB.

Tecnologies a utilitzar:

- Node.js
- **Biblioteca `ws`:** Per implementar WebSockets tant al servidor com al client (`npm install ws`).
- MongoDB (utilitzant la mateixa instància de Docker que a la PR3.1)
- Driver de MongoDB (`mongodb`).

Què cal fer?

- Crea una fitxa de requisits per la funcionalitat que es demana resoldre **(2.5 punts)**
- Crea una carpeta anomenada `pr32_websocket_link`
- Crea dos fitxers: `websocket_server.js` i `websocket_client.js` cal implementar aquesta funcionalitat **(5.5 punts)**:
  - El jugador s'ha de moure amb les fletxes.
  - El client enviarà la informació al servidor.
  - Sempre i quan el jugador estigui en moviment la informació de posició es considerarà la mateixa partida. Cada moviment en guardarà a MongoDB (1 document per moviment, però amb informació per associar-lo a la partida).
- Si el jugador passa 10 segons sense moure's es considera finalitzada la partida, el calcula la distància en línia recta entre el punt inicial i final i s'informa al client **(2 punts)**
- En els servidor cal implementar logs (pantalla i fitxer) amb winston.

Annex

Exemple fitxa de requisits

Sistema de Monitorització de Sensors en Temps Real:

Crea un servidor Node.js que utilitzi WebSockets per rebre lectures simulades de sensors (per exemple, temperatura i humitat) en format JSON des de múltiples clients Node.js. Emmagatzema cada lectura rebuda, juntament amb un timestamp, en una col·lecció de MongoDB. El servidor podria implementar una lògica per detectar lectures anòmales (fora d'un rang esperat) i notificar-ho als clients o registrar-ho de manera especial.

Fitxa de Requisits: Sistema de Monitorització de Sensors

Autor: [Marc Arqués Marimón] **Data:** [15/5/2025]

1. Objectiu Principal

Desenvolupar un sistema de registre de partides de joc 2D mitjançant WebSocket i Node.js, on un client envia els seus moviments al servidor. Aquest registra tots els moviments en MongoDB. Quan el jugador deixa de moure's durant 10 segons, la partida es dona per finalitzada i s'envia al client la distància total recorreguda en línia recta.

2. Requisits Funcionals (Què ha de fer)

Codi	Descripció
RF-01	El servidor WebSocket ha d'iniciar-se i escoltar connexions entrants.
RF-02	Els clients WebSocket han de poder connectar-se i rebre un missatge de benvinguda.
RF-03	El client ha de capturar moviments amb les fletxes i enviar-los com JSON.
RF-04	El servidor ha de registrar cada moviment com un document a MongoDB.
RF-05	Tots els moviments d'una mateixa partida han d'estar identificats com a grup.
RF-06	Si el jugador no es mou durant 10 segons, la partida finalitza automàticament.
RF-07	El servidor ha de calcular la distància entre el primer i l'últim moviment.
RF-08	El servidor ha d'enviar aquesta distància final al client.

3. Requisits No Funcionals (Com ho ha de fer)

Codi	Descripció
RNF-01	El servidor ha d'utilitzar `winston` per registrar esdeveniments (connexions, errors, missatges, etc.) tant per pantalla com a fitxer.
RNF-02	El codi ha d'estar comentat i ser fàcil de mantenir.
RNF-03	El sistema ha de gestionar errors sense tancar el servidor.
RNF-04	La connexió a MongoDB ha d'estar configurada amb paràmetres editables.

4. Format Missatge JSON (Client -> Servidor)

Els clients enviaran lectures amb l'estructura següent:

```
{
  "type": "move",
  "x": 10,
  "y": 15,
  "timestamp": 1686784354567
}
```
