🍼 IoT Baby Monitor Backend Roadmap (Node.js + TypeScript)

❓ سؤال البداية

هل قمت بإعداد MQTT Broker (Mosquitto) بالفعل، أم تحتاج مساعدة في ربط Node.js به؟

🧠 1. فهم النظام (Architecture)

النظام يتكون من 3 أجزاء رئيسية:

1️⃣ Hardware Layer

- ESP32 / Raspberry Pi
- يرسل بيانات الحساسات:
- Temperature
- Heart Rate
- Crying detection
- Position

📡 يرسل البيانات عبر MQTT Topics

2️⃣ MQTT Broker (Mosquitto)

- وسيط بين الأجهزة والـ Backend
- يستقبل البيانات من الحساسات
- يمررها إلى Node.js

📌 مثال Topics:

- baby/temperature
- baby/heartRate
- baby/crying

3️⃣ Backend (Node.js + TypeScript)

- يشترك في MQTT (Subscribe)
- يستقبل البيانات في الوقت الحقيقي
- يرسلها إلى Flutter عبر WebSocket

- يعالج التنبيهات (Alerts)

🔌 2. إعداد MQTT في Node.js

📦 تثبيت المكتبة

```
npm install mqtt
```

🧩 MQTT Service

```
import mqtt from "mqtt";

const client = mqtt.connect("mqtt://localhost:1883");

client.on("connect", () => {
  console.log("MQTT Connected");

  client.subscribe("baby/#", (err) => {
    if (!err) {
      console.log("Subscribed to baby topics");
    }
  });
});

client.on("message", (topic, message) => {
  const data = JSON.parse(message.toString());

  console.log("MQTT Data:", topic, data);

  // 👉 هنا تمرير البيانات للـ Business Logic أو Socket.io
});
```

🌐 3. إعداد WebSocket (Socket.io)

📦 تثبيت

```
npm install socket.io
```

🧩 Socket Server

```
import { Server } from "socket.io";

export const io = new Server(3001, {
  cors: {
    origin: "*",
  },
});

io.on("connection", (socket) => {
  console.log("Client connected:", socket.id);
});

export function emitBabyData(data: any) {
  io.emit("baby-data", data);
}
```

🔗 4. ربط MQTT مع Socket

```
import { emitBabyData } from "./socketServer";

client.on("message", (topic, message) => {
  const data = JSON.parse(message.toString());

  // 🔥 إرسال مباشر للتطبيق
  emitBabyData(data);
});
```

🚨 5. Alert Engine (تنبيهات)

```
function checkAlerts(data: any) {
  if (data.temperature >= 38.5) {
    console.log("🔥 FEVER ALERT");
  }

  if (data.isCrying) {
    console.log("👶 Baby is crying!");
  }
}
```

💾 6. Data Model (TypeScript)

```
export interface BabyHealthData {
  babyId: string;
  temperature: number;
  heartRate: number;
  position: 'on_back' | 'on_stomach';
  isCrying: boolean;
  timestamp: Date;
}
```

🧠 7. Business Logic Layer

- لا تعتمد على HTTP فقط
- البيانات تأتي كل ثانية
- يجب معالجة streaming data

🗄️ 8. تخزين البيانات (Best Practice)

❌ لا تخزن كل ثانية

✅ الأفضل:

- كل 1 دقيقة → average
- أو استخدام Time-Series DB

🚀 9. النظام الكامل

```
ESP32 → MQTT → Node.js → Socket.io → Flutter App
                    ↓
              Alert Engine
                    ↓
                Database
```

❓ السؤال التالي لك

هل تريد الآن:

1️⃣ إعداد Mosquitto (MQTT Broker) خطوة بخطوة على Ubuntu؟

2️⃣ أو ربط ESP32 مع MQTT؟

3️⃣ أو بناء Alert System متقدم (AI-like rules engine)؟