# Relatório Técnico – Fluxo MQTT e Dashboard IoT

Relatório – Fluxo MQTT + Configuração do Dashboard

## Objetivo e Escopo

Este protótipo demonstra um sistema de monitoramento cardíaco IoT com ESP32 gerando sinais vitais simulados e publicando em MQTT para consumo por Node-RED, onde uma Dashboard exibe:

Gráfico em tempo real de BPM (batimentos por minuto).

Gauge de temperatura corporal.

Alerta visual quando valores ultrapassam limiares clínicos (padrão: BPM > 120 e Temp > 38 °C).

A arquitetura reflete Fog/Edge + Cloud: parte da lógica e geração de dados ocorre na borda (ESP32); agregação, visualização e (opcionalmente) persistência/analytics rodam na nuvem.

## Arquitetura de Comunicação (MQTT)

Broker: HiveMQ Cloud (gerenciado, TLS 1.2, autenticação por usuário/senha).

Cliente Publicador (ESP32):

Conectividade via Wi-Fi.

TLS: WiFiClientSecure com setInsecure() para facilitar testes (em produção, usar CA adequada e validação de certificado).

Biblioteca PubSubClient: estabelece sessão, reconecta automaticamente e publica a cada 2 s.

Payload JSON com campos: device, ts, temp, hum, bpm, tht (threshold temp) e thb (threshold bpm).

Tópicos:

Publicação: cardio/telemetry/<deviceId> (telemetria).

Comandos/Config (opcional): cardio/cmd/<deviceId> para atualizar thresholds em tempo de execução (ex.: {"thresh\_temp":37.8}).

QoS, Retenção e Escalabilidade

QoS 0 suficiente para visualização em tempo real (baixa latência).

QoS 1 recomendado se houver persistência/alertas críticos.

Retained pode ser útil para último estado (não usamos aqui para evitar dados velhos no gráfico).

Escalabilidade por namespacing de tópicos e identidade por deviceId.

Segurança

TLS 8883 com autenticação.

Armazenar credenciais de forma segura (em hardware real, usar NVS ou partições seguras).

Em simulação, setInsecure() é didático; em produção, valide o servidor com CA.

## Geração dos Sinais (ESP32)

Temperatura/Umidade: DHT22 no pino 15. No Wokwi é possível “arrastar” a temperatura/humidade no componente para simular febre.

BPM: botão no pino 5 com interrupção. Contamos pressões em uma janela deslizante de 15s e multiplicamos para obter BPM aproximado (eventos/15s × 4). Debounce a 120 ms para evitar ruído.

Periodicidade: publicação a cada 2 s.

## Pipeline no Node-RED

MQTT IN: assina cardio/telemetry/# com TLS e credenciais do HiveMQ Cloud.

JSON parse: converte string → objeto.

Function (Extrai campos + alerta):

Separa temp (para gauge) e bpm (para gráfico).

Compara com tht/thb para definir alert.

Gera mensagens para três widgets: gauge (Temp), chart (BPM) e texto (status).

Dashboard (node-red-dashboard):

Gauge de Temperatura (min 30 – max 42).

Chart de BPM (40–180), janela visual de ~2 min.

Texto/Indicador: mostra “OK” ou mensagem de alerta; “bolinha” muda de cor (verde/vermelho).

Ajustes e Extensões

Trocar thresholds em tempo real via MQTT (publicando em cardio/cmd/<deviceId>).

Adicionar notificação (nó ui\_toast), e-mail ou webhook quando alerta disparar.

Persistir em InfluxDB e visualizar no Grafana para análises históricas (tendência de febre, episódios de taquicardia, etc.).

## Testes e Evidências

Conectividade: verificar logs no Serial (PUB: {...}) e no Node-RED (debug).

BPM: clicar rapidamente no botão por ~15 s; deve subir acima de 120 bpm e acionar alerta.

Temperatura: aumentar temperatura no DHT do Wokwi (> 38 °C) e observar gauge + alerta.

Estresse: manter o fluxo alguns minutos; checar estabilidade da conexão MQTT e renderização da UI.