**Sistema de Deteção de Incêndios com Alerta em Tempo Real**

José Guedes (a56576), Nelson Fernandes (a51796).

**Resumo:**

Este projeto propõe o desenvolvimento de um Sistema de Deteção de Incêndios com Alerta em Tempo Real, aplicado ao contexto de monitoramento ambiental em áreas de risco, como parques naturais. A solução é baseada na utilização de dispositivos IoT, nomeadamente o microcontrolador ESP32, equipado com sensores de temperatura e humidade (DHT22) e sensor de gases/fumaça (MQ-2). Em alternativa ao GPS, a aplicação OwnTracks foi utilizada para simular a localização dos dispositivos.  
Utilizando a plataforma Wokwi, simulou-se os dispositivos IoT, ESP32 e os sensores, que, quando detetam níveis anormais de temperatura ou gases, geram um alerta visual (LED) e sonoro (buzzer). As várias leituras de rotina são enviadas via MQTT para o Node-RED. As informações são então enviadas e guardadas numa base de dados InfluxDB e visualizadas através de uma dashboard construída no Node-RED e no Grafana (complementar), que apresenta os valores dos sensores, a localização do alerta (latitude e longitude) e o número de meios no terreno.  
Com esta solução, pretende-se demonstrar a viabilidade de um sistema distribuído e acessível para deteção precoce de incêndios, com capacidades de monitorização remota, registo histórico e visualização intuitiva através de uma interface gráfica.

**Palavras-chave**: Node-red, Incêndio, InfluxDB, IoT, ESP32.

**1. Introdução**

Nos últimos anos, os incêndios florestais tornaram-se cada vez mais frequentes, provocando perdas humanas, ambientais e financeiras. Por isso, é necessário reforçar e melhorar as capacidades de prevenção e deteção precoce de incêndios, de maneira a mitigar estes eventos. Uma das maneiras mais eficientes e práticas de mitigar estes eventos é recorrendo a dispositivos IoT. Os dispositivos IoT têm evoluído bastante nos últimos anos, tornando possível desenvolver sistemas de monitorização distribuídos, autónomos e de baixo custo, capazes de detetar em tempo real alterações ambientais que possam indicar o início de um incêndio.  
Este projeto propõe o desenvolvimento de um Sistema de Deteção de Incêndios com Alerta em Tempo Real, utilizando microcontroladores ESP32 e sensores específicos para recolha de dados ambientais, como temperatura, humidade e presença de gases. O sistema comunica através do protocolo MQTT, enviando a informação dos sensores para o Node-RED, juntamente com a aplicação OwnTracks, que permite simular a localização dos dispositivos. A informação recolhida é apresentada numa dashboard interativa, com integração da API dos fogos.pt, oferecendo uma visão abrangente e atualizada da situação no terreno. A simulação do sistema foi realizada na plataforma Wokwi, eliminando a necessidade de hardware físico durante a fase de desenvolvimento.

**2. Wokwi**

Começou-se por fazer as conexões dos dispositivos utilizados no ESP32. Como a figura 1 mostra.

Uma imagem com texto, captura de ecrã, eletrónica, círculo

Os conteúdos gerados por IA poderão estar incorretos.

Figura 1:Dispositivos no Wokwi

Seguiu-se o código do ESP32, foi utilizado o Broker: broker.emqx.io na porta TCP 1883. Num PC foi utilizado o tópico: guedes/alerta\_incendio e noutro nelson/alerta\_incendio, cada um simulando uma zona diferente.

A função setup\_wifi () é responsável pela conexão à rede WiFi e a função reconnect () pela conexão ao MQTT e a função setup () é responsável por inicializar a comunicação serial, configurar os pinos do LED e buzzer como saída, iniciar o sensor DHT22 e estabelecer a conexão com o WiFi e o servidor MQTT.

Já a função loop () é executada continuamente, verifica se a conexão MQTT está ativa, lê os valores de temperatura, humidade e gás, e determina se há risco de incêndio caso (temperatura > 50.0 || mq2\_analog > 1564) ativando o LED e o buzzer. Em seguida, o ESP32 envia uma mensagem em formato JSON com os dados lidos para o tópico MQTT correspondente. O facto de os valores já serem enviado no formato JSON facilita o processamento posterior no node-red.

Partes importantes/diferem do código Wokwi:

|  |
| --- |
| …  const char\* mqtt\_server = "broker.emqx.io";  const char\* mqtt\_topic = "guedes/alerta\_incendio"; OU const char\* mqtt\_topic = "nelson/alerta\_incendio";  void reconnect () {  …      String clientId = "ESP32-Fire-Guedes-"; OU String clientId = "ESP32-Fire-Nelson-";  …  }  void setup() {    …    client.setServer(mqtt\_server, 1883);  }  void loop() {   …    bool alerta = (temperatura > 50.0 || mq2\_analog > 1564); *// Valor 0.8 no sensor mq2*  …  *// Mensagem no formato JSON*    String mensagem = "{";    mensagem += "\"temperatura\":" + String(temperatura, 1) + ",";    mensagem += "\"humidade\":" + String(humidade, 1) + ",";    mensagem += "\"ppm\":" + String(mq2\_analog) + ",";    mensagem += "\"alerta\":" + String(alerta ? "true" : "false");    mensagem += "}";    Serial.println(mensagem);    client.publish(mqtt\_topic, mensagem.c\_str());  …  } |

**Todo o código deste projeto poderá ser encontrado em: https://github.com/FranciscoG08/Projeto\_IC**

**3. Node-red**

No Node-red começou-se a receber os dados via MQTT, e a mostrar na dashboard os valores para humidade, temperatura, gás e o estado do alerta, visto que no Wokwi os dados já eram enviados no formato JSON.

Para incorporar a localização utilizamos a aplicação Owntracks que permite enviar a localização do nosso telemóvel via MQTT. Para a localização utilizamos o tópico owntracks/batata/+, ou seja, vai receber mensagens publicadas nos subtópicos de owntracks/batata.  
A função *MAP* é importante porque é nela que associamos uma localização a cada utilizador, utilizamos a 3ª parte do tópico para saber quem é o utilizador (por exemplo, owntracks/batata/guedes, neste caso guedes é guardado como id), depois a localização latitude e longitude é guardada no objeto localizações, que é armazenada na memória do fluxo utilizando flow.set (). Isto é importante porque nas funções *Mapa\_Guedes* e *Mapa\_Nelson* utilizando flow.get () consegue-se aceder às localizações de cada um, utilizando apenas o id (guedes ou nelson), e com isso podemos então conectar essas funções ao *Map*, visto já se saber a localização de cada ESP32, agora na função verifica-se o estado do alerta e caso seja positivo aparecerá no mapa da dashboard o ícone de fogo na localização reportada.

Uma imagem com texto, captura de ecrã, diagrama, file

Os conteúdos gerados por IA poderão estar incorretos.

Figura 2: Implementação node-red comunicação via MQTT

Exemplo dos códigos das funções (os outros códigos são semelhantes, alterando apenas o sublinhado):

|  |  |
| --- | --- |
| Hum | msg.payload = msg.payload.humidade;  return msg; |
| Mapa\_Guedes | var dados = msg.payload;  var alerta = String(dados.alerta).toLowerCase() === "true";  var localizacoes = flow.get("localizacoes") || {};  var loc = localizacoes["guedes"] || { lat: 0, lon: 0 };  if (alerta) {      msg.payload = {          name: "incendio-guedes",          lat: loc.lat,          lon: loc.lon,          icon: "fa-fire",          layer: "incendios"      };  } else {      msg.payload = {          name: "incendio-guedes",          deleted: true,          layer: "incendios"      };  }  return msg; |
| MAP | var dados = msg.payload;  if (dados.\_type === "location") {      var id = msg.topic.split("/")[2]; *// Pode ter 2 valores*      var localizacoes = flow.get("localizacoes") || {};      localizacoes[id] = { lat: dados.lat, lon: dados.lon };      flow.set("localizacoes", localizacoes);  }  return null; |

Para melhorar o trabalho e torná-lo ainda mais completo foi implementada a API dos fogos.pt, assim através de um http request (Method: Get; URL: https://api.fogos.pt/new/fires) e convertendo os dados para o formato JSON, podemos incorporar no nosso mapa, utilizando a função Mapa API, os incêndios do nosso país. A API fornece várias informações, incluindo o estado de cada incêndio. Com base nesses dados, utilizámos uma estrutura *switch* para atribuir um ícone representativo a cada estado. Para tornar a visualização mais intuitiva, agrupámos alguns estados semelhantes sob o mesmo ícone e cor.

Além disso, foram adicionadas as funções Soma\_Bombeiros, Soma\_Terrestres, Soma\_Aerios, que apresentam o total de operacionais e meios no terreno. Foi ainda utilizada 3 templates para mostrar imagens na dashboard, uma para bombeiros, uma para meios terrestres e outra para meios aéreos.

Por fim, foi acrescentado a função Audio\_E\_PopUp, que sempre que se registar um novo incendio é informado via áudio a localização do novo incendio e através de um Pop-Up na dashboard.

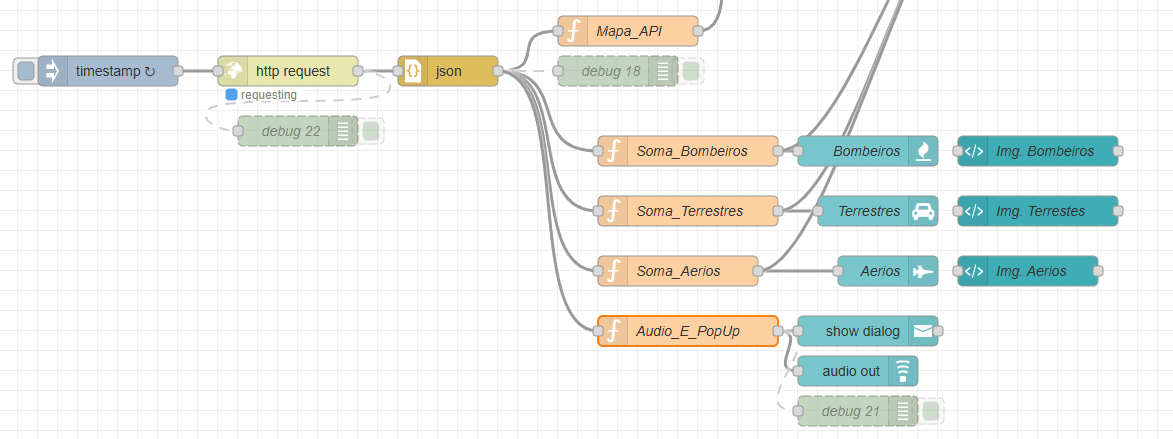


Figura 3: Implementação node-red http request

Para armazenar os dados foi usado o InfluxDB, neste projeto vamos armazenar os dados recolhidos dos 2 ESP’s, o número de bombeiros, meios terrestres e meios aéreos.

Na figura 4 podemos ver como ficou o fluxo no node-red.

Uma imagem com texto, diagrama, captura de ecrã, file

Os conteúdos gerados por IA poderão estar incorretos.

Figura 4: Fluxo final do Node-red

**3.1 Dashboard Node-red**

Resultados:

Uma imagem com texto, captura de ecrã, mapa, Software gráfico

Os conteúdos gerados por IA poderão estar incorretos.

Figura 5: Dashboard Node-red

Uma imagem com texto, captura de ecrã, diagrama, mapa

Os conteúdos gerados por IA poderão estar incorretos.

Figura 6: Dashboard Node-red PopUp

**4. InfluxDB**

No InfluxDB, armazenamos os dados recolhidos pelos ESP’s e o total de bombeiros, meios terrestre e aéreos.

Na figura 5 podemos ver as informações recolhidas pelos ESP’s.

Uma imagem com captura de ecrã, texto, Software de multimédia, software

Os conteúdos gerados por IA poderão estar incorretos.

Figura 7: InfluxDB informação dos ESP’s

Na figura 6 podemos ver o gráfico com a evolução do número de meios no local.

Uma imagem com captura de ecrã, Software de multimédia

Os conteúdos gerados por IA poderão estar incorretos.

Figura 8: InfluxDB informação dos Meios

**5. Grafana (extra)**

No Grafana foi criada uma dashboard diferente com as informações mais relevantes recolhidas do InfluxDB.

Uma imagem com captura de ecrã, diagrama, Modelagem 3D, astronomia

Os conteúdos gerados por IA poderão estar incorretos.

Figura 9: Grafana Dashboard

**5. Conclusões**

Este projeto demonstra a utilidade dos dispositivos IoT na prevenção e combate a incêndios. A utilização do ESP32, juntamente com sensores ambientais e comunicação MQTT, permitiu criar uma solução escalável e de fácil implementação, com alertas automáticos em caso de risco de incêndio.

A integração com o Node-RED possibilitou a criação de uma dashboard bastante informativa e de fácil leitura, enquanto a utilização do InfluxDB e do Grafana garantiu o armazenamento de dados e análise histórica.

A utilização da plataforma OwnTracks e da API dos fogos.pt enriqueceu o sistema com funcionalidades de localização e acesso a dados reais sobre incêndios ativos em Portugal, oferecendo ao utilizador uma visão mais completa da situação. A simulação no Wokwi permitiu desenvolver o sistema sem recorrer a equipamentos físicos. Para concluir, o trabalho demonstra o potencial das tecnologias IoT na prevenção e gestão de desastres naturais, oferecendo uma aplicação de baixo custo, com grande aplicabilidade prática em contextos de monitorização ambiental e proteção civil.

**Referências Bibliográficas**

[1] Associação Fogos.pt. (2023). *API dos Fogos – Informação sobre Incêndios em Portugal*. Associação Fogos.pt, Portugal.

[2] Grafana Portugal (2023). *Visualização de Dados em Tempo Real com Grafana e InfluxDB*. Comunidade de Código Aberto, Porto, Portugal.

[3] OwnTracks. (2023). *Location – OwnTracks Booklet*.

[4] EMQX. (2023). *Public MQTT Broker Documentation*.

[5] FlowFuse. (2023). *MQTT – Node-RED Core Nodes Documentation*. FlowFuse.

[6] FlowFuse. (2023). *InfluxDB – Node-RED Database Integration*. FlowFuse.