|  |  |  |
| --- | --- | --- |
|  | UNIVERSIDADE FEDERAL DE SANTA CATARINA Programa de Pós-Graduação em Engenharia de *Automação e Sistemas* | |
| DATA | | Outubro de 2020. |
| DISCIPLINA | | Técnicas de Implementação de Sistemas Automatizados |
| PROFESSORES | | Carlos Montez e Leandro Buss |
| ALUNOS | | Matuzalem Müller  Michela Limaco  Patrícia Mayer |

**Monitoramento de Temperatura e Umidade com sensor DHT11**

O sistema monitora a temperatura e umidade com o sensor DHT11, utiliza o MQTT para receber o comando de iniciar e parar a aquisição dos dados e armazena os dados processados no banco NoSQL Firebase. A interface de comunicação com o usuário foi desenvolvida utilizando o Node-RED. O usuário delimita uma faixa de temperatura desejada e quando a temperatura coletada estiver fora da faixa o sistema notifica o usuário por e-mail. A seguir descrevemos em mais detalhes a construção do sistema.

Utilizamos o microcontrolador ESP8266 com o sensor de DHT11 para adquirir os dados de Temperatura e Umidade. A cada 1000 aquisições os dados são processados utilizando-se uma biblioteca de cálculos estatísticos para extrair os valores de média, mínimo, máximo, variância e desvio padrão. Adicionalmente foi implementando um algoritmo de Machine Learning para determinar se o dado lido corresponde à um outlier, caso seja essa informação também é armazenada no Firebase.

A Figura 1 exibe o código da inicialização do software, onde é extraído o MAC Address do WiFi do microcontrolador, em seguida é estabelecida a conexão com a rede WiFi, os vetores estatísticos que vão armazenar as leituras são inicializados e o modelo do Machine Learning é carregado para memória EEPROM.

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Figura 1 – Setup do ESP8266

Dentro do método que conecta o uControlador à WiFi também ocorre a conexão ao broker MQTT, a conexão com o Firebase e a sincronização do RTC interno com um servidor ntp.

O método Loop, que equivale à rotina principal do programa, verifica se o MQTT está conectado, e fica ‘escutando’ o tópico que está assinando. O tópico que o MQTT assina corresponde ao comando de start/stop na aquisição dos dados. Quando o valor é de START (‘1’), uma variável booleana lerSensor recebe TRUE, enquanto lerSensor for true o sistema fará aquisição dos dados continuamente, como ilustrado na Figura 2.

Texto

Descrição gerada automaticamente

Figura 2 – Loop do ESP8266

Tabela

Descrição gerada automaticamenteA aquisição e processamento dos dados do DHT11 ocorre no método readSensor, exibido na Figura3.

Figura 3 – readSensor do ESP8266

O valor de temperatura e umidade é lido do DHT11, adicionado ao vetor estatístico e a cada 1000 aquisições são extraídos valores de média, mínimo, máximo, variância e desvio padrão. O valor é processado também pela função SVM\_predictEEPROM que retorna se o valor é um outlier ou não. Os dados são encapsulados em um json e armazenados no Firebase. Na Figura 4 observa-se o dado armazenado no Firebase.

Figura 4 – Informação de Temperatura no Firebase

Como mencionamos no início o uControlador recebe comandos via MQTT, utiliza o node-RED para interface com o usuário e utiliza o serviço de envio de e-mail para notificação. Tanto node-RED quanto o serviço de envio de e-mails estão hospedados em um servidor na nuvem.

Interface gráfica do usuário, Tabela

Descrição gerada automaticamenteUtilizamos o Google Cloud Platform para o servidor em nuvem, na primeira tentativa utilizamos tentamos rodar também o broker mqtt mas a VM apresentava problemas de desempenho, a opção foi separar o broker e aumentar a memória da VM, na Figura 5 vemos a configuração da VM dentro do gerenciados do GCP. A configuração final selecionada foi g1-small (1 vCPU, 1.7 GB memória).

Figura 5 – Configuração da VM dentro do GCP

No Namecheap foi registro um domínio próprio para acessar o servidor em nuvem – https://nodered.matuzalemmuller.com, e um certificado SSL foi gerado gratuitamente utilizando o certbot do Let's Encrypt: https://letsencrypt.org/.

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente Dessa forma configuramos regras de firewall para permitir o acesso através da porta 443 e uma regra de firewall interna para conexão via SSH e monitoramento da VM. A Figura 6 exibe as regras cridas na GCP.

Figura 6 – Regras de Firewall na GCP

Com o servidor acessível via URL pública foi necessário configurar uma autenticação para evitar que pessoas não autorizadas pudessem ter acesso ao Node-RED. Para habilitar autenticação no Node-RED, foi necessário alterar as linhas 119-126 do arquivo settings.js (da instalação do Node-RED) para habilitar autenticação e definir as credenciais de acesso:

adminAuth: {

type: "credentials",

users: [{

username: "admin",

password: "$2b$08$eUiTa1NvNya0FQPAH6EHherx5BIahq6/k1qSfbNIf.FKxsloHY$

permissions: "\*"

}]

},

Texto

Descrição gerada automaticamenteA estrutura dos arquivos dentro do servidor em nuvem é exibida na Figura 7.

Figura 7 – Arquivos do servidor GCP

O Build de imagens é realizado através do arquivo docker-compose.yml (docker-compose-build). As imagens para o Node-RED e Flask são criadas através de seus Dockerfiles correspondentes, sendo que a imagem do Flask é criada a partir de uma imagem oficial do python para linux alpine e o Dockerfile do Node-RED é fornecido pela própria equipe do Node-RED em <https://github.com/node-red/node-red-docker>. Flask é container web para Python que executa o serviço de enviar e-mails, requerido para notificação do usuário quando a temperatura está fora da faixa configurada.

Para habilitar HTTPS no Node-RED, é preciso alterar as configurações das linhas 141-144 e 165:

https: {

key: require("fs").readFileSync('/app/privkey.pem'),

cert: require("fs").readFileSync('/app/fullchain.pem')

},

requireHttps: true,

As bibliotecas de dashboard e do Firebase também são automaticamente instaladas ao iniciar o contêiner, através do arquivo package.json:

"dependencies": {

"node-red": "1.2.1",

"node-red-contrib-firebase" : "1.1.1",

"node-red-dashboard" : "2.23.4"

}

Conforme mencionado o Node-RED é executado no servidor em nuvem descrito acima. Vamos explicar o funcionamento do fluxo no Node-RED, onde são representados os itens do dashboard no qual, ao clicar no botão START é enviado o parâmetro ‘1’ para o tópico dto/dht11/status, o qual o ESP8266 está subscrito. Quando o ESP8266 recebe '1' ele começa a coletar os dados dos sensores e envia para o banco NoSQL. Ao clicar no STOP, é enviado ‘0’ para o mesmo tópico e o ESP8266 interrompe a coleta dos dados. Também é apresentado uma mensagem de status para o usuário, como apresenta a Figura 8.

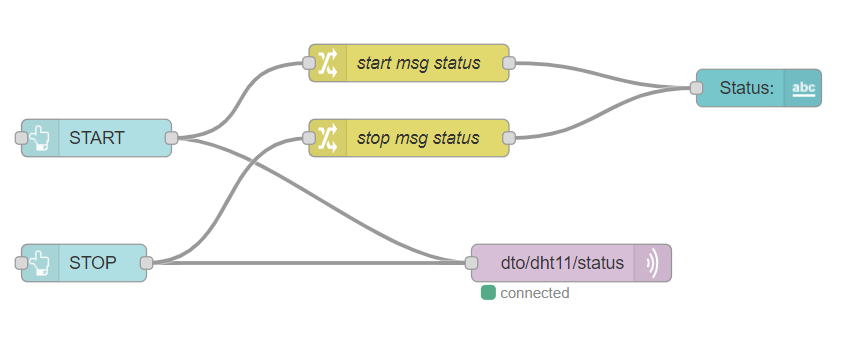


Figura 8 – Fluxo dos nodos de START e STOP

Os valores médios de temperatura e umidade são apresentados em formato de gráfico e de medidor, respectivamente, como exibe a Figura 9.

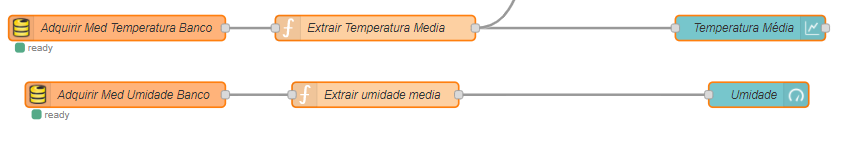
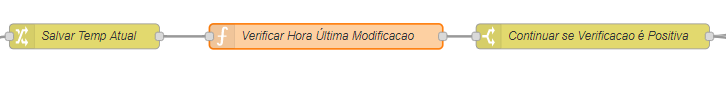


Figura 9 – Nodos de exibição da temperatura e da umidade

A partir dos valores de temperatura lidos, a temperatura atual é alocada em uma variável global e um *timestamp* é introduzido para se ter a relação da hora atual com a hora de notificação para o usuário via e-mail a cada 5 minutos. Com isso, realiza-se a comparação com o tempo atual para verificar se decorreram os minutos desejados, como mostra a Figura 10.

Figura 10 – Verificação de tempo decorrido

Com isso, pode-se comparar as temperaturas máximas e mínimas obtidas com as quais foram requisitadas pelo usuário, caso o valor esteja fora da faixa, este é transformado em *string* a qual é convertida em um .json para encaminhar a notificação para o usuário de acordo com a aplicação *flask*. Por fim, a última hora é apresentada para efeitos de comparação, apresentado nas Figura 11 e Figura 12.

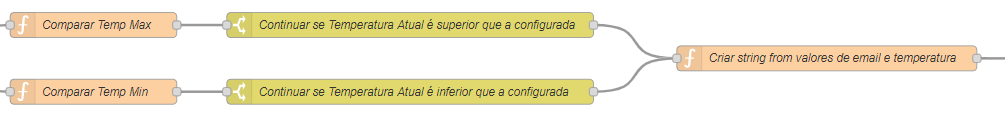


Figura 11 – Comparativo de temperaturas máximas e mínimas



Figura 12 – Criação do .json, notificação para o usuário e arquivamento da última hora de notificação

Uma imagem contendo Interface gráfica do usuário

Descrição gerada automaticamenteA Figura 14 exibe o Dashboard gerado pelo Node-red, acessível pelo browser através do endereço https://nodered.matuzalemmuller.com/ui.

Figura 14 – Dashboard Web construída com paleta ‘Dashboard’ gerada pelo Node-RED

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamentePara concluir na Figura 15 é exibido o diagrama de funcionamento da solução de monitoramento de Temperatura e Umidade.

Figura 15 – Diagrama de Funcionamento do Sistema de Monitoramento de Temperatura e Umidade