Implantação do Escritório Inteligente

1. Instalação da plataforma Dojot

   Requerimentos de hardware:
   - CPU: 4 núcleos
   - RAM: 4GB
   - Espaço livre em disco: 10GB

   As seguintes portas devem estar abertas:
   - TCP (conexões de entrada): 1883 (MQTT), 8883 (MQTT seguro, se utilizado), 8000 (acesso à interface web)

   Dependências:
   - Docker Engine: https://docs.docker.com/engine/install/
   - Docker-compose: https://docs.docker.com/compose/install/

   Instalação da Dojot v0.4.2:

   Para construir o ambiente, clone o repositório e execute os comandos abaixo.

```
git clone https://github.com/dojot/docker-compose.git
# Let's move into the repo - all commands in this page should be executed
# inside it.
cd docker-compose
git checkout v0.4.2 -b v0.4.2
# Must be run from the root of the deployment repo.
# May need sudo to work: sudo docker-compose up -d
docker-compose up -d
# Shows the list of currently running containers, along with individual info
docker ps

# Shows the list of all configured containers, along with individual info
docker ps -a
```

   Para parar o serviço execute o comando:

```
docker-compose down
```

   Obs.: O guia completo de instalação da Dojot pode ser encontrado na documentação do projeto: https://dojotdocs.readthedocs.io/pt_BR/stable/installation-guide.html

2. Cadastro de recursos na Dojot

Após a instalação, acessando a dashboard da Dojot (localhost:8000, usuario: admin, senha: admin), é possível importar o arquivo de configuração para cadastro de recursos na plataforma acessando **configurações > import./exportação > exportação** e selecionando o arquivo **DojotData.json** localizado no diretório **EscritorioInteligente/** em **Dojot/Data/** .

Para que dispositivos físicos enviem mensagens à Dojot, é necessária a criação de dispositivos virtuais na plataforma. Além disso, a Dojot permite a criação de workflows que processam os dados enviados pelos dispositivos e a partir disso executam determinada ação.

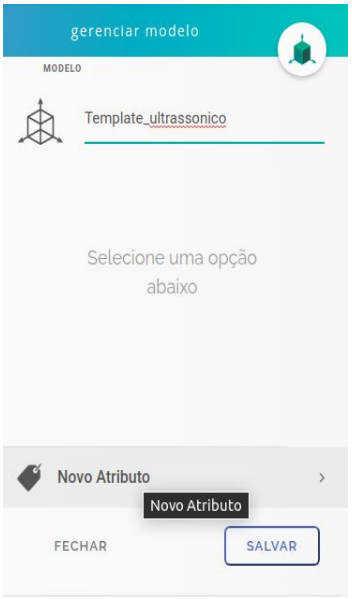
Na Dojot, um dispositivo é a representação virtual de dispositivos reais com um ou mais sensores. Todo dispositivo é criado com base em um Template que, por sua vez, contém um rótulo e uma lista de atributos. A seguir é descrito um passo a passo para a criação de templates, dispositivos e workflows utilizando a dashboard da plataforma.

Criação de um template para sensor ultrassônico:

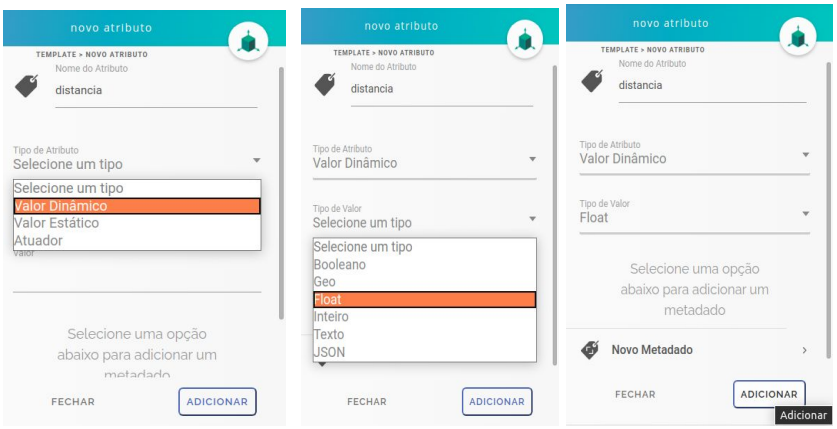
No menu lateral esquerdo selecione a opção ‘Modelos’.

Em seguida selecione ‘Novo modelo’ para iniciar a criação de um novo template.

Especifique o nome do template e selecione a opção ‘Novo atributo’ para adicionar atributos ao modelo.



Defina as especificações do atributo: tipo de atributo e tipo de valor a ser armazenado no atributo. Para adicionar o atributo selecione a opção ‘Adicionar’.



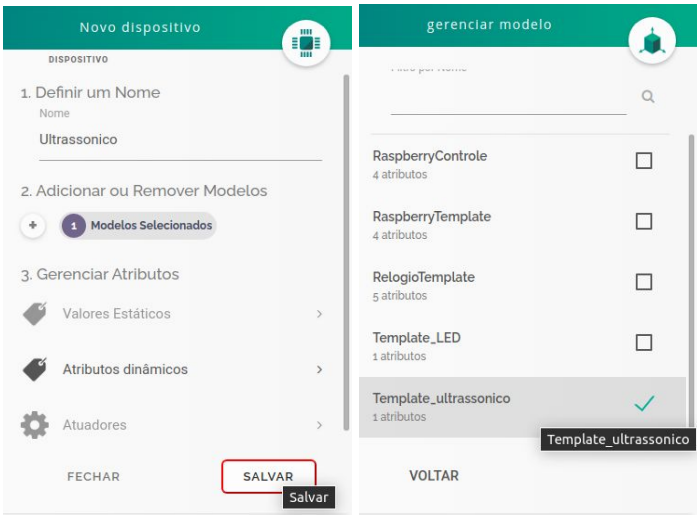
Após adicionados todos os atributos, finalize a criação do novo template clicando na opção ‘Salvar’.

Criação de um novo dispositivo:

No menu lateral esquerdo selecione a opção ‘Dispositivos’.

Em seguida selecione ‘Novo dispositivo’ para iniciar a criação de um novo dispositivo.

Especifique o nome, assim como os templates que irão compor o novo dispositivo. Após realizadas as alterações necessárias, finalize selecionado a opção ‘Salvar’.



Criação de workflow:

A seguir é descrito o passo a passo para a criação de um novo workflow na plataforma Dojot. O objetivo do fluxo criado é acender um LED com base na leitura

de distância realizada pelo sensor ultrassônico. Para isso, foi criado um novo dispositivo nomeado LED, contendo apenas um atributo ‘Booleano’ do tipo ‘Atuador’ para especificar se a lâmpada está ou não está acesa.

No menu lateral esquerdo selecione a opção ‘Fluxos’.

Em seguida selecione ‘Novo fluxo’ para iniciar a criação de um novo fluxo.

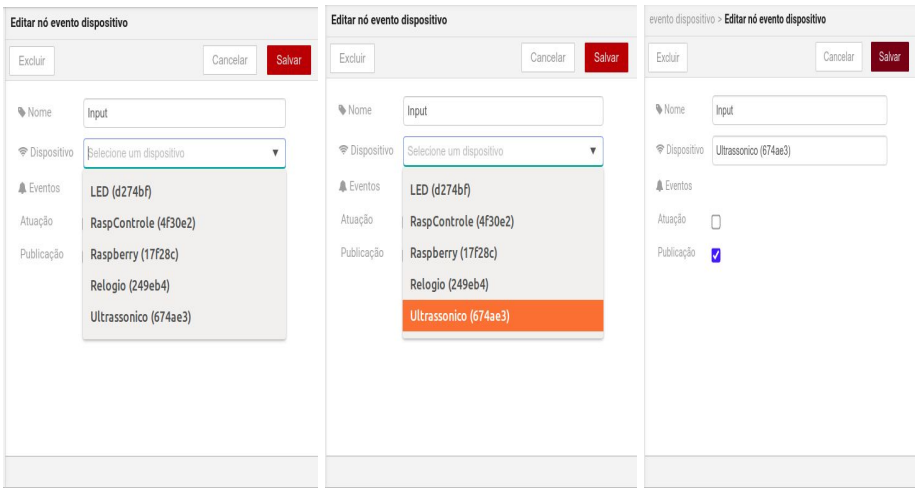
Na barra lateral esquerda aparecerá uma lista de blocos que realizam o processamento de mensagens e eventos de dispositivos. Para adicionar um bloco ao novo fluxo, basta selecionar e arrastar o bloco desejado para o centro da tela.

Um fluxo necessariamente contém:

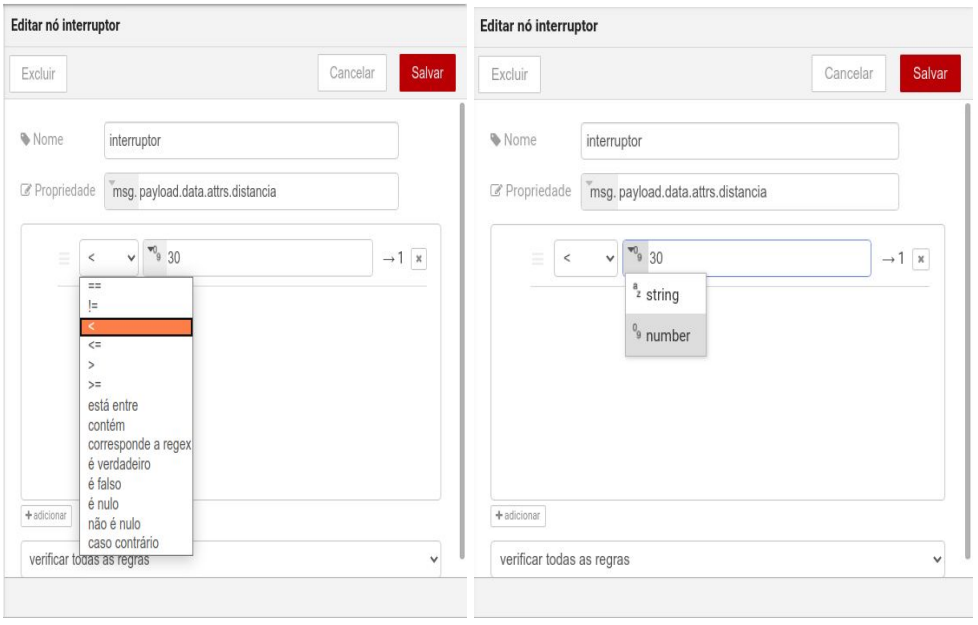
- Ponto de entrada: gatilho para iniciar o fluxo específico;
- Bloco de processamento: conjunto de blocos que executar operações com base no evento recebido;

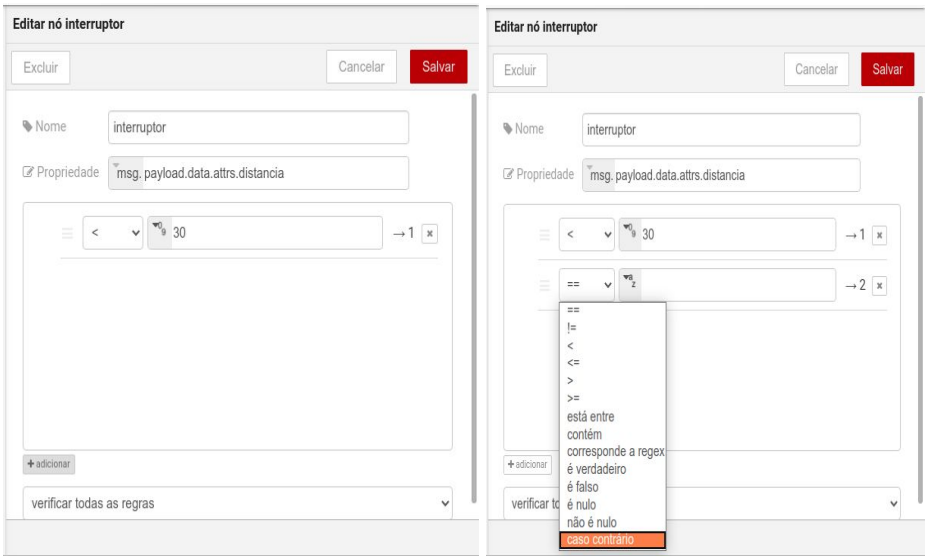
- Ponto de saída: para onde os resultados devem ser encaminhados

Para iniciar a criação do novo fluxo, selecione o bloco ‘evento dispositivo’ para ser o ponto de entrada. Em seguida, defina as configurações do bloco: nome, o dispositivo que acionará o fluxo (nesse caso, o sensor ultrassônico) e quais eventos irão acionar o fluxo. Finalize selecionando a opção ‘Salvar’.

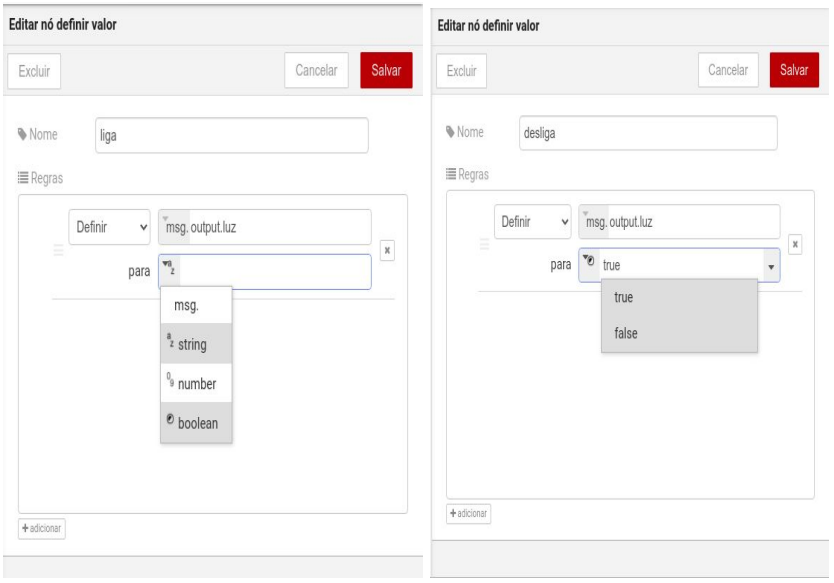


Para configurar o bloco de processamento do workflow, selecione o bloco ‘Interruptor’, que realiza a análise de dados recebidos do dispositivo de entrada. Após isso, especifique as configurações do nó interruptor adicionando nome e as condições para tomada de decisão. Nesse caso, a decisão será tomada com base nos dados recebidos pelo atributo ‘distancia’ do dispositivo que engatilhou o fluxo.

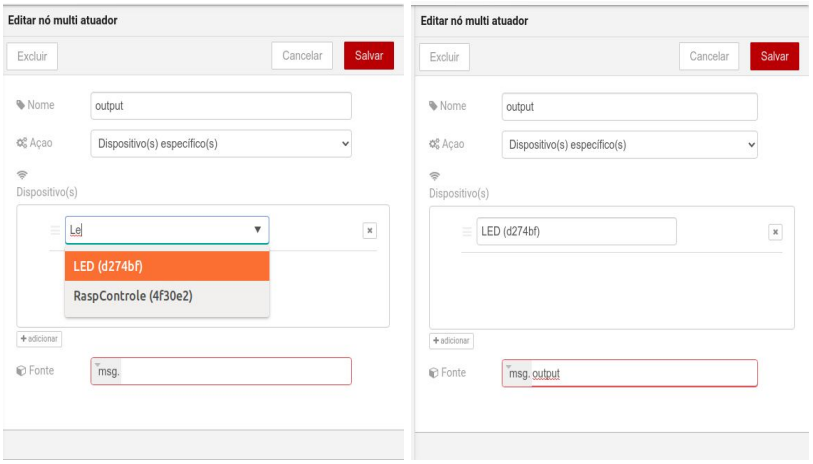




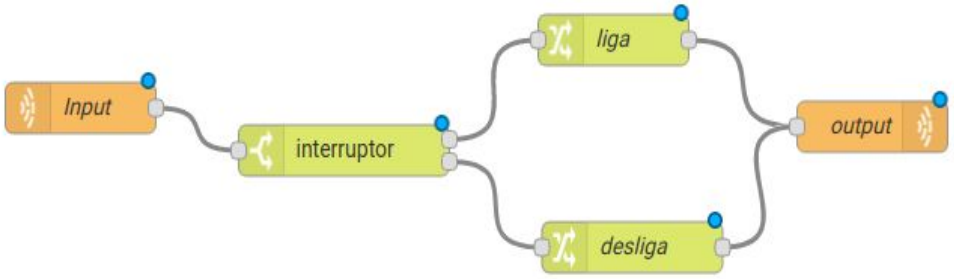
Ainda na etapa de processamento, selecione os blocos que serão acionados para cada caso. Para isso, escolha suas unidades do bloco ‘Definir valor’ (um para acender a lâmpada e outro para desligá-la) e indique os ajustes de cada bloco. O bloco intitulado ‘liga’ altera o valor do atributo ‘luz’ do dispositivo de saída para ‘true’ e o bloco desliga realiza a alteração para ‘false’.



Feito isso, selecione o bloco ‘Multi atuado’ para ser o ponto de saída. Em suas configurações defina o nome e o dispositivo de saída.



Até aqui foram selecionados todos os blocos necessários para a criação do fluxo. Para finalizar, realize a conexão dos blocos ligando todos os conectores da seguinte forma:



Especifique o nome do novo fluxo e clique em ‘Salvar’ para concluir a criação do novo workflow.

Com o fluxo criado, todos os dados recebidos pelo dispositivo ‘Ultrassônico’ serão analisados e, caso a distância lida seja menor do que trinta centímetros, uma mensagem de valor ‘true’ será enviada ao dispositivo ‘LED’. Da mesma forma, caso a distância lida seja maior ou igual a trinta, o dispositivo ‘LED’ recebe uma mensagem de valor ‘false’.

A documentação oficial da Dojot traz de forma detalhada as características e funcionalidades de cada bloco do construtor de fluxos: https://dojotdocs.readthedocs.io/pt_BR/stable/flow.html.

3. Instalação da plataforma FIWARE

As seguintes portas devem estar abertas:

- 1883 (MQTT), 1026 (orion context broker), 27017 (MongoDB), 4041 (IoT Agent)

Dependências:

- Docker Engine: https://docs.docker.com/engine/install/
- Docker-compose: https://docs.docker.com/compose/install/

Instalação da Fiware:
Para construir o ambiente, execute os comandos abaixo:

```
sudo docker pull mongo:3.6
sudo docker pull fiware/orion
git clone https://github.com/FIWARE/tutorials.IoT-over-MQTT.git
cd tutorials.IoT-over-MQTT
sudo ./services create
sudo ./services start
```

Para parar o serviço execute o comando:

```
sudo ./services stop
```

Obs.: O guia completo de instalação pode ser encontrado na documentação do projeto: https://fiware-tutorials.readthedocs.io/en/latest/getting-started/index.html. A documentação do IoT Agent MQTT pode ser acessada em https://fiware-tutorials.readthedocs.io/en/latest/iot-over-mqtt/index.html.

4. Cadastro de recursos na FIWARE

Após a instalação, é necessário cadastrar na plataforma 'entidades de dados de contexto' de forma a possibilitar a comunicação entre dispositivos físicos e o context broker. No diretório **EscritoInteligente** em **Fiware/Data** está localizado o arquivo **data_fiware.sh** que realiza o cadastro automático desses recursos na Fiware. Execute os seguintes comandos no terminal:

```
chmod +x data_fiware.sh
./data_fiware.sh
```

A documentação do IoT Agent MQTT (https://fiware-tutorials.readthedocs.io/en/latest/iot-over-mqtt/index.html#connecting-iot-devices) apresenta detalhadamente o processo de criação de dispositivos na Plataforma Fiware.

5. Configuração do ambiente para execução da aplicação Escritório Inteligente

5.1 Instalação de bibliotecas necessárias para a execução da aplicação de e-mails:

a) Paho MQTT:
Versão utilizada: paho-mqtt (1.5.0)
Comando de instalação: `pip3 install paho-mqtt`
Documentação: https://pypi.org/project/paho-mqtt/

b) APScheduler:
Versão utilizada: APScheduler (3.6.3)
Comando de instalação: `pip3 install apscheduler`

Documentação: https://apscheduler.readthedocs.io/en/stable/

5.2 Instalação de bibliotecas necessárias para a execução da aplicação relógio:

a) Paho MQTT:
Versão utilizada: paho-mqtt (1.5.0)
Comando de instalação: `pip3 install paho-mqtt`
Documentação: https://pypi.org/project/paho-mqtt/

b) APScheduler:
Versão utilizada: APScheduler (3.6.3)
Comando de instalação: `pip3 install apscheduler`
Documentação: https://apscheduler.readthedocs.io/en/stable/

c) Requests:
Versão utilizada: requests (2.18.4)
Comando de instalação: `pip3 install requests`
Documentação: https://pypi.org/project/requests/

5.3 Instalação de bibliotecas necessárias para a execução da aplicação de monitoramento nos nós. As seguintes bibliotecas devem ser instaladas em cada Raspberry Pi:

a) Paho MQTT:
Versão utilizada: paho-mqtt (1.5.0)
Comando de instalação: `pip3 install paho-mqtt`
Documentação: https://pypi.org/project/paho-mqtt/

b) APScheduler:
Versão utilizada: APScheduler (3.6.3)
Comando de instalação: `pip3 install apscheduler`
Documentação: https://apscheduler.readthedocs.io/en/stable/

c) RPi.GPIO
Versão utilizada: RPi.GPIO (0.7.0)
Comando de instalação: `pip3 install RPi.GPIO`
Documentação: https://pypi.org/project/RPi.GPIO/

5.4 Instalação de bibliotecas necessárias para a execução da API REST:

a) Flask
Versão utilizada: Flask (1.1.2)
Comando de instalação: `pip3 install flask`
Documentação: https://flask.palletsprojects.com/en/1.1.x/installation/

b) PyMongo

Versão Utilizada: pymongo (3.11.0)
Comando de instalação: `pip3 install pymongo`
Documentação: https://api.mongodb.com/python/current/tutorial.html

5.5 Clonagem de repositório

Execute o comando `git clone …` para clonar o repositório.

Para realizar as configurações necessárias, selecione dentro do diretório **EscritorioInteligente** a pasta que contém o nome da plataforma escolhida para implantação (Dojot ou Fiware).

É importante mencionar que a pasta **Raspberry** contém a aplicação de monitoramento que deverá ser executada em cada nó Raspberry Pi. Sendo assim, cada um dos nós deverá conter uma cópia do diretório **EscritorioInteligente/*Plataforma*/Raspberry**.

5.6 Alterações necessárias no arquivo de configuração da aplicação de e-mail:

A seguir são listadas modificações a serem realizadas no arquivo **config.yaml** localizado no diretório **EscritorioInteligente/** em **Dojot/Monitoring/Scripts**.

Especificações do escritório a ser monitorado:

```
room:
      id: X
      nodes: X
```

O campo ‘id’ corresponde ao ID da sala no sistema de agendamentos da RNP.

O parâmetro ‘nodes’ corresponde a quantidade de nós configurados para realizar o monitoramento da sala X.

Especificações de e-mails e alertas:

```
email:
      email_sender: user1@email.com
      password_sender: pass123
      email_receiver: user2@email.com
      availability:
            subject: Sub1
            message: msg1
      failure:
            subject: Sub2
            message: msg2
```

Os campos ‘email_sender’ e ‘password_sender’ correspondem aos dados da conta de e-mail responsável pelo envio de mensagens e alertas ao administrador do ambiente.

O campo ‘email_receiver’ refere-se ao endereço de e-mail do administrador do ambiente, no caso, o responsável por receber informações a respeito da disponibilidade da sala e eventuais falhas apresentadas pelos sensores.

O campo ‘avalibility’ contém os dados do e-mail a ser enviado em caso do ambiente ficar disponível antes do fim de determinada reserva. ‘subject’ refere-se ao assunto do e-mail e ‘mensagem’ refere-se ao conteúdo da mensagem a ser enviada.

O campo ‘failure’ contém os dados do e-mail a ser enviado em caso de um ou mais nós responsáveis pelo monitoramento da sala apresentar alguma falha de funcionamento. O parâmetro ‘subject’ refere-se ao assunto do e-mail e ‘message’ refere-se ao conteúdo da mensagem a ser enviada. É importante ressaltar que junto a mensagem especificada pelo administrador, serão enviados dados referentes ao número de identificação do nó assim como o horário em que sua última mensagem de atividade normal foi registrada.

Especificações do Broker MQTT:

```
mqtt_broker:
      host: 127.0.0.1
      port: 1883
```

O campo ‘host’ refere-se ao IP do broker MQTT, ou seja, o IP da máquina onde foi instalado IoT Agent MQTT da plataforma Dojot.

O campo ‘port’ contém a porta para conexão com o broker MQTT, que, por padrão, é a porta 1883.

5.7 Arquivo de configuração da aplicação relógio

A seguir são listadas modificações a serem realizadas no arquivo **config.yaml** localizado no diretório **EscritorioInteligente/** em **Dojot/Scheduler/Scripts**.

Especificações de usuário para acesso a api de agendamentos da RNP:

```
scheduling_api:
      user: user@email.com
      password: pass123
```

Configurações de intervalos de tempo para cancelamento da reserva e frequência de consultas a API do sistema de agendamento:

```
set_times:
      query_database: X
      cancel_booking: X
```

O campo ‘query_database’ refere-se à frequência (em minutos) em que a aplicação relógio realiza consultas à API do sistema de agendamento da RNP.

Assim que uma determinada reserva se inicia, a aplicação de e-mail verifica se, durante o período de alocação, a sala ficará vazia por um intervalo de tempo (em minutos) superior ao especificado no campo ‘cancel_booking’.

Especificações do Broker MQTT:

```
mqtt_broker:
      host: 127.0.0.1
      port: 1883
```

O campo ‘host’ refere-se ao IP do broker MQTT, ou seja, o IP da máquina onde foi instalado IoT Agent MQTT da plataforma Dojot.

O campo ‘port’ contém a porta para conexão com o broker MQTT, que, por padrão, é a porta 1883.

5.8 Arquivo de configuração dos nós da sala

Cada nó possui seu arquivo de configuração contendo algumas especificações próprias. O arquivo **config.yaml** contendo as configurações da aplicação de monitoramento encontra-se no diretório **Raspberry/Scripts** pertencente a cada um dos Raspberries.

Especificações do nó:

```
node:
      node_id: X
      distance_max: X
```

O campo 'node_id' corresponde ao ID do nó em que o arquivo de configuração se encontra. O ID deve ser um número inteiro.

O campo 'distance_max' refere-se a distância (em centímetros) entre o nó e seu primeiro obstáculo fixo localizado no ambiente, como por exemplo a distância entre o nó e uma parede que esteja a sua frente. A detecção de presença é realizada com base na variação desse intervalo.

Especificações do Broker MQTT:

```
mqtt_broker:
      host: 127.0.0.1
      port: 1883
```

O campo 'host' refere-se ao IP do broker MQTT, ou seja, o IP da máquina onde foi instalado IoT Agent MQTT da plataforma Dojot.

O campo 'port' contém a porta para conexão com o broker MQTT, que, por padrão, é a porta 1883.

6. Execução das aplicações e acompanhamento de Logs

Após preencher os parâmetros de cada arquivo de configuração, execute os seguintes comandos no terminal

O arquivo docker-compose para executar o container contendo o MongoDB (docker-compose.yml) está localizado dentro do diretorio **EscritorioInteligente/**.

Para executar a API Rest, dentro do diretório **EscritorioInteligente/api_rest/** escute o comando `nohup python3 run.py`.

O arquivo de execução da aplicação de e-mail (send_email.py) está localizado dentro do diretório **EscritorioInteligente/** em ***Plataforma*/Monitoring/Code**. O comando `nohup python3 send_email.py &` permite que a aplicação rode em segundo plano.

O arquivo de logs da aplicação de e-mail (logs_monitoring.log) está localizado dentro no diretório **EscritorioInteligente/** em ***Plataforma*/Monitoring/Logs**. O comando `tail -f logs_monitoring.log` permite o acompanhamento em tempo real dos logs gerados pela aplicação de e-mail.

O arquivo de execução da aplicação relógio (run.py) está localizado dentro do diretório **EscritorioInteligente/** em ***Plataforma*/Scheduler/Code**. O comando `nohup python3 run.py &` permite que a aplicação rode em segundo plano.

O arquivo de logs da aplicação relógio (logs_scheduler.log) está localizado dentro no diretório **EscritorioInteligente/** em ***Plataforma*/Scheduler/Logs**. O comando `tail -f`

`logs_scheduler.log` permite o acompanhamento em tempo real dos logs gerados pela aplicação relógio.

O arquivo de execução da aplicação de monitoramento dos nós (sensor_ultrassonico.py) está localizado dentro do diretório **Raspberry/Code**. O comando `nohup python3 sensor_ultrassonico.py &` permite que a aplicação rode em segundo plano. Lembre-se de realizar esse procedimento em todos os nós.

O arquivo de logs da aplicação de monitoramento (logs.log) está localizado dentro no diretório **Raspberry/Logs**. O comando `tail -f logs.log` permite o acompanhamento em tempo real dos logs gerados pelo monitoramento.