

**PONTIFÍCIA UNIVERSIDADE CATÓLICA DO PARANÁ**

**CIÊNCIA DA COMPUTAÇÃO.**

PJBL4 – ENTREGA FINAL

Bruno teider, FERNANDO MORO, LUIZ MATHIAS FILHO, NICOLAS ANDREAS JACKEL, RAFAELA VECCHI

**CURITIBA**

**2024**

**SUMÁRIO**

[1 INTRODUÇÃO: Descrição do Projeto 4](#_Toc168649666)

[2 WEB 4](#_Toc168649667)

[2.1 Controladores e visualizações 4](#_Toc168649668)

[2.2 Página Inicial 26](#_Toc168649669)

[2.3 Dados Históricos 26](#_Toc168649670)

[2.4 Flask Login e Cadastro 27](#_Toc168649671)

[2.5 Cadastro e listagem sensores e atuadores 28](#_Toc168649672)

[3 Banco de Dados 30](#_Toc168649673)

[3.1 Modelos SQLAlchemy 30](#_Toc168649674)

[3.2 Métodos de inserir, selecionar, editar e excluir 32](#_Toc168649675)

[4 Hardware 35](#_Toc168649676)

[4.1 Esquemático do Hardware 35](#_Toc168649677)

[4.2 Tópicos MQTT utilizados 35](#_Toc168649678)

[5 Integração 36](#_Toc168649679)

[5.1 Integração Web, Banco de Dados e Hardare 36](#_Toc168649680)

[5.2 Comunicação IoT Flask-Mqtt 37](#_Toc168649681)

# 1 INTRODUÇÃO: Descrição do Projeto

O Nicolabs é um projeto que tem como objetivo ter um maior controle laboratorial, ele realiza a medição de temperatura, umidade e nível de gás e com esses dados, pode acionar um alarme caso haja algum tipo de risco. O projeto foi desenvolvido utilizando Esp32 no site Wokwi, Flask, Mqtt e banco de dados MySql.

# 2 WEB

## 2.1 Controladores e visualizações

AppController.py:

from flask import Flask

from flask\_mqtt import Mqtt

from database.config import db, uri

from models import Leitura

from models.dadosModel import Dados

# Inicialização do MQTT

mqtt\_client = Mqtt()

# Lista de tópicos para inscrição

topicos\_inscritos = [

    "bflnr/valorUmidade",

    "bflnr/valorTemperatura",

    "bflnr/valorGas",

]

def create\_app():

    app = Flask(

        \_\_name\_\_,

        template\_folder="./views/",

        static\_folder="./static/",

        root\_path="./"

    )

    app.config['SECRET\_KEY'] = 'secret-key'

    app.config["SQLALCHEMY\_DATABASE\_URI"] = uri

    db.init\_app(app)

    # Configurações do MQTT

    app.config['MQTT\_BROKER\_URL'] = 'mqtt-dashboard.com'

    app.config['MQTT\_BROKER\_PORT'] = 1883

    app.config['MQTT\_KEEPALIVE'] = 5000  # KeepAlive time in seconds

    # Inicializa o MQTT no contexto da aplicação

    mqtt\_client.init\_app(app)

    # Registra os blueprints

    from controllers.indexController import index\_bp

    app.register\_blueprint(index\_bp, url\_prefix='/')

    from controllers.loginController import login\_bp

    app.register\_blueprint(login\_bp, url\_prefix='/auth')

    # Define os callbacks do MQTT

    define\_mqtt\_callbacks(app)

    return app

def define\_mqtt\_callbacks(app):

    @mqtt\_client.on\_connect()

    def handle\_connect(client, userdata, flags, rc):

        if rc == 0:

            print('Broker Conectado com sucesso')

            for topic in topicos\_inscritos:

                mqtt\_client.subscribe(topic)  # Se inscreve nos tópicos

        else:

            print('Bad connection. Code:', rc)

    @mqtt\_client.on\_disconnect()

    def handle\_disconnect(client, userdata, rc):

        print("Broker desconectado")

    @mqtt\_client.on\_message()

    def handle\_mqtt\_message(client, userdata, message):

        with app.app\_context():

            payload = message.payload.decode("utf-8")  # Decodifica o payload como string

            topic = str(message.topic)

            sensor = ""

            print("\*\*\*\*\*\*\* ", topic, payload)

            if topic == "bflnr/valorUmidade":

                Dados['umidade'] = float(payload)

                sensor = "Umidade"

            elif topic == "bflnr/valorTemperatura":

                Dados['temperatura'] = float(payload)

                sensor = "Temperatura"

            elif topic == "bflnr/valorGas":

                Dados['gas'] = float(payload)

                sensor = "Gás"

            else:

                return  # Ignora tópicos não reconhecidos

            nova\_leitura = Leitura(sensor=sensor, medicao=payload)

            db.session.add(nova\_leitura)

            db.session.commit()

def send\_mqtt\_message(topic, message):

    mqtt\_client.publish(topic, message)

indexController.py:

from flask import Blueprint, render\_template, session, redirect, url\_for, request

from database.config import db

from models import \*

from controllers.appController import send\_mqtt\_message

index\_bp = Blueprint("index", \_\_name\_\_)

@index\_bp.route('/')

def index():

  if 'email' in session:

    led = Atuador.query.filter\_by(apelido="LED").first()

    buzzer = Atuador.query.filter\_by(apelido="Buzzer").first()

    return render\_template("pages/home/index.html", temperatura=Dados['temperatura'], umidade=Dados['umidade'], gas=Dados['gas'], led=led, buzzer=buzzer)

  else:

    return redirect(url\_for('login.login\_index'))

# SENSORES ------------------------------------------------------------------------------------------------------------

@index\_bp.route('/sensores')

def sensores():

  sensores = Sensor.query.all()

  total = len(sensores)

  sensores\_ativos = Sensor.query.filter\_by(ativo=True).all()

  totalAtivos = len(sensores\_ativos)

  if session['admin'] != True and session['operador'] != True:

    return redirect(url\_for('index.index'))

  else:

    return render\_template("pages/sensores/sensores.html", sensores=sensores, total=total, totalAtivos=totalAtivos)

@index\_bp.route('/criaSensor', methods=['POST'])

def criarSensor():

  nome = request.form['nomeAddSensor']

  medicao = request.form['medicaoAddSensor']

  ativo = request.form['ativoAddSensor'] == 'True'

  novo\_sensor = Sensor(apelido=nome, medicao=medicao, ativo=ativo)

  db.session.add(novo\_sensor)

  db.session.commit()

  return redirect(url\_for('index.sensores'))

@index\_bp.route('/editarSensor', methods=['POST'])

def editarSensor():

  nome = request.form['nomeEditarSensor']

  medicao = request.form['medicaoEditarSensor']

  ativo = request.form['ativoEditarSensor'] == 'True'

  \_idSensor = request.form['idEditarSensor']

  sensor = Sensor.query.get(\_idSensor)

  if sensor:

      sensor.apelido = nome

      sensor.medicao = medicao

      sensor.ativo = ativo

      db.session.commit()

  return redirect(url\_for('index.sensores'))

@index\_bp.route('/removerSensor/<\_id>')

def removerSensor(\_id):

  sensor = Sensor.query.get(\_id)

  if sensor:

      db.session.delete(sensor)

      db.session.commit()

  return redirect(url\_for('index.sensores'))

# ATUADORES --------------------------------------------------------------------------------------------------------------

@index\_bp.route('/atuadores')

def atuadores():

  atuadores = Atuador.query.all()

  total = len(atuadores)

  atuadores\_ativos = Atuador.query.filter\_by(ativo=True).all()

  totalAtivos = len(atuadores\_ativos)

  if session['admin'] != True and session['operador'] != True:

    return redirect(url\_for('index.index'))

  else:

    return render\_template("pages/atuadores/atuadores.html", atuadores=atuadores, total=total, totalAtivos=totalAtivos)

@index\_bp.route('/criaAtuador', methods=['POST'])

def criarAtuador():

  nome = request.form['nomeAddAtuador']

  ativo = request.form['ativoAddAtuador'] == 'True'

  tipo = request.form['tipoAddAtuador']

  novo\_atuador = Atuador(apelido=nome, ativo=ativo, tipo=tipo)

  db.session.add(novo\_atuador)

  db.session.commit()

  return redirect(url\_for('index.atuadores'))

@index\_bp.route('/editarAtuador', methods=['POST'])

def editarAtuador():

  nome = request.form['nomeEditarAtuador']

  ativo = request.form['ativoEditarAtuador'] == 'True'

  \_idAtuador = request.form['idEditarAtuador']

  atuador = Atuador.query.get(\_idAtuador)

  if atuador:

      atuador.apelido = nome

      atuador.ativo = ativo

      db.session.commit()

  return redirect(url\_for('index.atuadores'))

@index\_bp.route('/removerAtuador/<\_id>')

def removerAtuador(\_id):

  atuador = Atuador.query.get(\_id)

  if atuador:

    db.session.delete(atuador)

    db.session.commit()

  return redirect(url\_for('index.atuadores'))

# USUÁRIOS --------------------------------------------------------------------------------------------------------------

@index\_bp.route('/usuarios')

def usuarios():

  users = Users.query.all()

  total = len(users)

  users\_admins = Users.query.filter\_by(admin=True).all()

  totalAdmins = len(users\_admins)

  if session['admin'] != True:

    return redirect(url\_for('index.index'))

  else:

    return render\_template("pages/usuarios/usuarios.html", users=users, total=total, totalAdmins=totalAdmins)

# LEITURAS --------------------------------------------------------------------------------------------------------------

@index\_bp.route('/leituras')

def leituras():

  leituras = Leitura.query.all()

  # Formata a data para exibição

  for leitura in leituras:

      leitura.data = leitura.data.strftime("%H:%M %d/%m/%Y")

  if session['admin'] != True:

    return redirect(url\_for('index.index'))

  else:

    return render\_template("pages/leituras/leituras.html", leituras=leituras)

# CONTROLES REMOTOS ------------------------------------------------------------------------------------------------------

@index\_bp.route('/acionaLed')

def acionaLed():

  valor = '0'

  atuador = Atuador.query.filter\_by(apelido="LED").first()

  atuador.ativo = atuador.ativo != True # INVERTE O VALOR DO SENSOR

  db.session.commit()

  if atuador.ativo:

    valor = '1'

  else:

    valor = '0'

  send\_mqtt\_message('bflnr/valorLed', valor)

  return redirect(url\_for('index.index'))

@index\_bp.route('/acionaBuzzer')

def acionaBuzzer():

  send\_mqtt\_message('bflnr/valorBuzzer', '1')

  return '', 204

loginController.py:

from flask import Blueprint, render\_template, request, redirect, url\_for, session

from database.config import db

from models import Users

login\_bp = Blueprint("login", \_\_name\_\_)

@login\_bp.route('/')

def login\_index():

  return render\_template("pages/login/login.html")

@login\_bp.route('/postLogin', methods=['POST'])

def postLogin():

    email = request.form['emailLogin']

    senha = request.form['senhaLogin']

    autenticado = False

    user = Users.query.filter\_by(email=email).first()

    if user:

      if user.senha == senha:

        session['email'] = user.email

        session['nome'] = user.name

        session['admin'] = user.admin

        session['operador'] = user.operador

        autenticado = True

    if autenticado:

      return redirect(url\_for("index.index"))

    else:

      return render\_template("pages/login/login.html", error='ERRO NA AUTENTICAÇÃO')

@login\_bp.route('/criaUsuario', methods=['POST'])

def criarUsuario():

  email = request.form['emailAddUser']

  nome = request.form['nomeAddUser']

  senha = request.form['senhaAddUser']

  admin = request.form['adminAddUser'] == 'True'

  operador = request.form['operadorAddUser'] == 'True'

  novo\_user = Users(name=nome, email=email, senha=senha, admin=admin, operador=operador)

  db.session.add(novo\_user)

  db.session.commit()

  return redirect(url\_for('index.usuarios'))

@login\_bp.route('/editarUsuario', methods=['POST'])

def editarUsuario():

  email = request.form['emailEditarUser']

  nome = request.form['nomeEditarUser']

  senha = request.form['senhaEditarUser']

  admin = request.form['adminEditarUser'] == 'True'

  operador = request.form['operadorEditarUser'] == 'True'

  \_id = request.form['idEditarUser']

  user = Users.query.get(\_id)

  if user:

      user.name = nome

      user.senha = senha

      user.email = email

      user.admin = admin

      user.operador = operador

      db.session.commit()

  return redirect(url\_for('index.usuarios'))

@login\_bp.route('/removerUsuario/<\_id>')

def removerUsuario(\_id):

  user = Users.query.get(\_id)

  if user and not user.admin:

    db.session.delete(user)

    db.session.commit()

  return redirect(url\_for('index.usuarios'))

@login\_bp.route('/logout')

def logout():

  session.clear()

  return redirect(url\_for('login.login\_index'))

atuadores.html:

{% extends 'base.html' %}

{% block body %}

    {% include '\_includes/sideBar.html'%}

    <section class="page">

        <div class="banner">

            <h2>Atuadores</h2>

        </div>

        <div class="atuadores--container">

            <div class="atuadores--dados">

                <!-- ATUADORES TOTAIS -->

                <div class="atuadores--dados\_\_card">

                    <h3>Total de Atuadores</h3>

                    <div>

                        <i class="material-icons">lightbulb</i>

                        <p>{{ total }}</p>

                    </div>

                </div>

                <!-- ATUADORES ATIVOS -->

                <div class="atuadores--dados\_\_card">

                    <h3>Atuadores Ativos</h3>

                    <div>

                        <i class="material-icons">tips\_and\_updates</i>

                        <p>{{ totalAtivos }}</p>

                    </div>

                </div>

                <!-- ADD ATUADOR -->

                <div class="atuadores--dados\_\_card cardAddAtuador" onclick="open\_popupAddAtuador()">

                    <h3>Adicionar Atuador</h3>

                    <div>

                        <i class="material-icons">add\_circle\_outline</i>

                    </div>

                </div>

            </div>

            <div class="atuadores--listagem">

                {% for atuador in atuadores %}

                    <div class="atuadores--listagem\_\_atuadorCard">

                        <div class="atuadorInfos">

                            <i class="material-icons">{% if atuador.tipo == 'LED' %}lightbulb\_circle{% else %}volume\_up{% endif %}</i>

                            <div class="line"></div>

                            <h3>{{ atuador.apelido }}</h3>

                            <p>{{ atuador.medicao }}</p>

                            <span>{% if atuador.ativo %} <span class="ativo">Ativo</span> {%else%} <span class="atuador">Inativo</span> {% endif %}</span>

                        </div>

                        <div class="atuadorOptions">

                            <i class="material-icons edit" onclick="open\_popupEditarAtuador('{{ atuador.id }}', '{{ atuador.apelido}}', '{{ atuador.ativo }}')">edit</i>

                            <a href="/removerAtuador/{{ atuador.id }}"><i class="material-icons delete">delete</i></a>

                        </div>

                    </div>

                {% endfor %}

            </div>

        </div>

    </section>

    <!-- POPUP ADD SENSOR -->

    <dialog id="modal--addAtuador">

        <div class="container">

            <form action="/criaAtuador" method="POST">

                <legend>Adicionar atuador</legend>

                <div class="inputContainer">

                    <label for="nomeAddAtuador">Nome</label>

                    <input type="text" name="nomeAddAtuador" id="nomeAddAtuador" required>

                </div>

                <div class="inputContainer">

                    <label for="tipoAddAtuador">Tipo</label>

                    <select id="tipoAddAtuador" name="tipoAddAtuador">

                        <option value="LED" selected>LED</option>

                        <option value="Buzzer">Buzzer</option>

                    </select>

                </div>

                <div class="inputContainer">

                    <label for="ativoAddAtuador">Ativo</label>

                    <select id="ativoAddAtuador" name="ativoAddAtuador">

                        <option value="True" selected>Sim</option>

                        <option value="False">Não</option>

                    </select>

                </div>

                <div class="btns">

                    <button class="btnCriar" type="submit">Salvar</button>

                    <button class="btnVoltar" type="button" onclick="closeModal\_popupAddAtuador()">Cancelar</button>

                </div>

            </form>

        </div>

    </dialog>

    <!-- POPUP EDITAR SENSOR -->

    <dialog id="modal--editarAtuador">

        <div class="container">

            <form action="/editarAtuador" method="POST">

                <legend>Editar atuador</legend>

                <input type="hidden" name="idEditarAtuador" id="idEditarAtuador">

                <div class="inputContainer">

                    <label for="nomeEditarAtuador">Nome</label>

                    <input type="text" name="nomeEditarAtuador" id="nomeEditarAtuador" required>

                </div>

                <div class="inputContainer">

                    <label for="ativoEditarAtuador">Ativo</label>

                    <select id="ativoEditaAtuador" name="ativoEditarAtuador">

                        <option value="True" selected>Sim</option>

                        <option value="False">Não</option>

                    </select>

                </div>

                <div class="btns">

                    <button class="btnEditar" type="submit">Editar</button>

                    <button class="btnVoltar" type="button" onclick="closeModal\_popupEditarAtuador()">Cancelar</button>

                </div>

            </form>

        </div>

    </dialog>

{% endblock %}

index.html:

{% extends 'base.html' %}

{% block meta %}

<meta http-equiv="refresh" content="3">

{% endblock %}

{% block body %}

    {% include '\_includes/sideBar.html'%}

    <section class="page">

        <div class="banner">

            <h2>Meu laboratório</h2>

        </div>

        <div class="dados">

            <!-- TEMPERATURA -->

            <div id="temperatura" class="dados--card">

                <h3>Temperatura</h3>

                <div>

                    <i class="material-icons">device\_thermostat</i>

                    <p>{{ temperatura }}°C</p>

                </div>

                <i class="material-icons">tune</i>

            </div>

            <!-- HUMIDADE -->

            <div id="umidade" class="dados--card">

                <h3>Humidade</h3>

                <div>

                    <i class="material-icons">water\_drop</i>

                    <p>{{ umidade }}%</p>

                </div>

                <i class="material-icons">tune</i>

            </div>

            <!-- GAS -->

            <div id="gas" class="dados--card">

                <h3>Gás GLP</h3>

                <div>

                    <i class="material-icons">propane\_tank</i>

                    <p>{{ gas }}ppm</p>

                </div>

                <i class="material-icons">tune</i>

            </div>

        </div>

        <div class="controlesRemotos">

            {% if led %}

            <div class="controleRemoto--card">

                <h3>Acionar Led:</h3>

                <form action="/acionaLed" method="GET">

                    <button class="{% if led.ativo == True%}ativo{%endif%}" type="submit"><i class="material-icons">lightbulb</i></button>

                </form>

            </div>

            {% endif %}

            {% if buzzer %}

            <div class="controleRemoto--card">

                <h3>Acionar Buzzer:</h3>

                <form action="/acionaBuzzer" method="GET">

                    <button type="submit"><i class="material-icons">volume\_up</i></button>

                </form>

            </div>

            {% endif %}

        </div>

    </section>

{% endblock %}

Leituras.html:

{% extends 'base.html' %}

{% block body %}

    {% include '\_includes/sideBar.html'%}

    <section class="page">

        <div class="banner">

            <h2>Leituras</h2>

        </div>

        <div class="usuarios--container">

            <div class="usuarios--listagem">

                {% for leitura in leituras %}

                    <div class="usuarios--listagem\_\_userCard">

                        <div class="userInfos">

                            <i class="material-icons">settings\_remote</i>

                            <div class="line"></div>

                            <h3>{{leitura.sensor}}</h3>

                            <p style="flex-grow: 1;">{{leitura.medicao}}</p>

                            <div class="line"></div>

                            <p style="text-align: end;">{{leitura.data}}</p>

                        </div>

                    </div>

                {% endfor %}

            </div>

        </div>

    </section>

{% endblock %}

login.html:

{% extends 'base.html' %}

{% block body %}

<div class="authBg">

  <div class="container">

      <div class="authBanner">

          <img src="/static/img/LOGO02.webp" alt="Logo">

      </div>

  </div>

  <div class="container authForms">

      <form class="authForms--signin" action="/auth/postLogin" method="POST">

          <legend>Entrar</legend>

          <div class="inputContainer">

              <label for="emailLogin">Email</label>

              <input type="email" name="emailLogin" id="emailLogin" required>

          </div>

          <div class="inputContainer">

              <label for="senhaLogin">Senha</label>

              <input type="password" name="senhaLogin" id="senhaLogin" required>

          </div>

          <button type="submit" class="buttonLogin">Logar</button>

          {% if error %}

            <div class="msg error">{{error}}</div>

          {% endif %}

      </form>

  </div>

</div>

{% endblock %}

sensores.html:

{% extends 'base.html' %}

{% block body %}

    {% include '\_includes/sideBar.html'%}

    <section class="page">

        <div class="banner">

            <h2>Sensores</h2>

        </div>

        <div class="sensores--container">

            <div class="sensores--dados">

                <!-- SENSORES TOTAIS -->

                <div class="sensores--dados\_\_card">

                    <h3>Total de Sensores</h3>

                    <div>

                        <i class="material-icons">sensors</i>

                        <p>{{ total }}</p>

                    </div>

                </div>

                <!-- SENSORES ATIVOS -->

                <div class="sensores--dados\_\_card">

                    <h3>Sensores Ativos</h3>

                    <div>

                        <i class="material-icons">online\_prediction</i>

                        <p>{{ totalAtivos }}</p>

                    </div>

                </div>

                <!-- ADD SENSOR -->

                <div class="sensores--dados\_\_card cardAddSensor" onclick="open\_popupAddSensor()">

                    <h3>Adicionar Sensor</h3>

                    <div>

                        <i class="material-icons">add\_circle\_outline</i>

                    </div>

                </div>

            </div>

            <div class="sensores--listagem">

                {% for sensor in sensores %}

                    <div class="sensores--listagem\_\_sensorCard">

                        <div class="sensorInfos">

                            <i class="material-icons">contactless</i>

                            <div class="line"></div>

                            <h3>{{sensor.apelido}}</h3>

                            <p>{{sensor.medicao}}</p>

                            <span>{% if sensor.ativo == True %} <span class="ativo">Ativo</span> {%else%} <span class="sensor">Inativo</span> {% endif %}</span>

                        </div>

                        <div class="sensorOptions">

                            <i class="material-icons edit" onclick="open\_popupEditarSensor('{{ sensor.id }}', '{{ sensor.apelido }}', '{{ sensor.medicao }}', '{{ sensor.ativo }}')">edit</i>

                            <a href="/removerSensor/{{ sensor.id }}"><i class="material-icons delete">delete</i></a>

                        </div>

                    </div>

                {% endfor %}

            </div>

        </div>

    </section>

    <!-- POPUP ADD SENSOR -->

    <dialog id="modal--addSensor">

        <div class="container">

            <form action="/criaSensor" method="POST">

                <legend>Adicionar sensor</legend>

                <div class="inputContainer">

                    <label for="nomeAddSensor">Nome</label>

                    <input type="text" name="nomeAddSensor" id="nomeAddSensor" required>

                </div>

                <div class="inputContainer">

                    <label for="medicaoAddSensor">Medição</label>

                    <input type="text" name="medicaoAddSensor" id="medicaoAddSensor" required>

                </div>

                <div class="inputContainer">

                    <label for="ativoAddSensor">Ativo</label>

                    <select id="ativoAddSensor" name="ativoAddSensor">

                        <option value="True" selected>Sim</option>

                        <option value="False">Não</option>

                    </select>

                </div>

                <div class="btns">

                    <button class="btnCriar" type="submit">Salvar</button>

                    <button class="btnVoltar" type="button" onclick="closeModal\_popupAddSensor()">Cancelar</button>

                </div>

            </form>

        </div>

    </dialog>

    <!-- POPUP EDITAR SENSOR -->

    <dialog id="modal--editarSensor">

        <div class="container">

            <form action="/editarSensor" method="POST">

                <legend>Editar usensor</legend>

                <input type="hidden" name="idEditarSensor" id="idEditarSensor">

                <div class="inputContainer">

                    <label for="nomeEditarSensor">Nome</label>

                    <input type="text" name="nomeEditarSensor" id="nomeEditarSensor" required>

                </div>

                <div class="inputContainer">

                    <label for="medicaoEditarSensor">Medição</label>

                    <input type="text" name="medicaoEditarSensor" id="medicaoEditarSensor" required>

                </div>

                <div class="inputContainer">

                    <label for="ativoEditarSensor">Ativo</label>

                    <select id="ativoEditarSensor" name="ativoEditarSensor">

                        <option value="True" selected>Sim</option>

                        <option value="False">Não</option>

                    </select>

                </div>

                <div class="btns">

                    <button class="btnEditar" type="submit">Editar</button>

                    <button class="btnVoltar" type="button" onclick="closeModal\_popupEditarSensor()">Cancelar</button>

                </div>

            </form>

        </div>

    </dialog>

{% endblock %}

usuarios.html:

{% extends 'base.html' %}

{% block body %}

    {% include '\_includes/sideBar.html'%}

    <section class="page">

        <div class="banner">

            <h2>Usuários</h2>

        </div>

        <div class="usuarios--container">

            <div class="usuarios--dados">

                <!-- USUARIOS TOTAIS -->

                <div class="usuarios--dados\_\_card">

                    <h3>Total de Usuários</h3>

                    <div>

                        <i class="material-icons">group</i>

                        <p>{{ total }}</p>

                    </div>

                </div>

                <!-- USUARIOS ADMINISTRADORES -->

                <div class="usuarios--dados\_\_card">

                    <h3>Usuarios Administradores</h3>

                    <div>

                        <i class="material-icons">admin\_panel\_settings</i>

                        <p>{{ totalAdmins }}</p>

                    </div>

                </div>

                <!-- ADD USUÁRIO -->

                <div class="usuarios--dados\_\_card cardAddUsuario" onclick="open\_popupAddUser()">

                    <h3>Adicionar Usuário</h3>

                    <div>

                        <i class="material-icons">person\_add</i>

                    </div>

                </div>

            </div>

            <div class="usuarios--listagem">

                {% for user in users %}

                    <div class="usuarios--listagem\_\_userCard">

                        <div class="userInfos">

                            <i class="material-icons">portrait</i>

                            <div class="line"></div>

                            <h3>{{user.name}}</h3>

                            <p>{{user.email}}</p>

                            <span>{% if user.admin %} <span class="admin">Administrador</span> {%else%} <span class="user">Usuario</span> {% endif %}</span>

                            <span>{% if user.operador %} <span class="operador">Operador</span>{% endif %}</span>

                        </div>

                        <div class="userOptions">

                            <i class="material-icons edit" onclick="open\_popupEditarUser('{{ user.id }}', '{{ user.email }}', '{{ user.name }}', '{{ user.senha }}', '{{ user.admin }}', '{{ user.operador }}')">edit</i>

                            <a href="/auth/removerUsuario/{{ user.id }}"><i class="material-icons delete">delete</i></a>

                        </div>

                    </div>

                {% endfor %}

            </div>

        </div>

    </section>

    <!-- POPUP ADD USUÁRIO -->

    <dialog id="modal--addUsuario">

        <div class="container">

            <form action="/auth/criaUsuario" method="POST">

                <legend>Criar usuário</legend>

                <div class="inputContainer">

                    <label for="emailAddUser">Email</label>

                    <input type="email" name="emailAddUser" id="emailAddUser" required>

                </div>

                <div class="inputContainer">

                    <label for="nomeAddUser">Nome</label>

                    <input type="text" name="nomeAddUser" id="nomeAddUser" required>

                </div>

                <div class="inputContainer">

                    <label for="senhaAddUser">Senha</label>

                    <input type="password" name="senhaAddUser" id="senhaAddUser" required>

                </div>

                <div class="inputContainer">

                    <label for="adminAddUser">Administrador</label>

                    <select id="adminAddUser" name="adminAddUser">

                        <option value="True">Sim</option>

                        <option value="False" selected>Não</option>

                    </select>

                </div>

                <div class="inputContainer">

                    <label for="operadorAddUser">Operador</label>

                    <select id="operadorAddUser" name="operadorAddUser">

                        <option value="True">Sim</option>

                        <option value="False" selected>Não</option>

                    </select>

                </div>

                <div class="btns">

                    <button class="btnCriar" type="submit">Salvar</button>

                    <button class="btnVoltar" type="button" onclick="closeModal\_popupAddUser()">Cancelar</button>

                </div>

            </form>

        </div>

    </dialog>

    <!-- POPUP EDITAR USUÁRIO -->

    <dialog id="modal--editarUsuario">

        <div class="container">

            <form action="/auth/editarUsuario" method="POST">

                <legend>Editar usuário</legend>

                <input type="hidden" name="idEditarUser" id="idEditarUser">

                <div class="inputContainer">

                    <label for="emailEditarUser">Email</label>

                    <input type="email" name="emailEditarUser" id="emailEditarUser" required>

                </div>

                <div class="inputContainer">

                    <label for="nomeEditarUser">Nome</label>

                    <input type="text" name="nomeEditarUser" id="nomeEditarUser" required>

                </div>

                <div class="inputContainer">

                    <label for="senhaEditarUser">Senha</label>

                    <input type="password" name="senhaEditarUser" id="senhaEditarUser" required>

                </div>

                <div class="inputContainer">

                    <label for="adminEditarUser">Administrador</label>

                    <select id="adminEditarUser" name="adminEditarUser">

                        <option value="True">Sim</option>

                        <option value="False" selected>Não</option>

                    </select>

                </div>

                <div class="inputContainer">

                    <label for="operadorEditarUser">Operador</label>

                    <select id="operadorEditarUser" name="operadorEditarUser">

                        <option value="True">Sim</option>

                        <option value="False" selected>Não</option>

                    </select>

                </div>

                <div class="btns">

                    <button class="btnEditar" type="submit">Editar</button>

                    <button class="btnVoltar" type="button" onclick="closeModal\_popupEditarUser()">Cancelar</button>

                </div>

            </form>

        </div>

    </dialog>

{% endblock %}

base.html:

<!DOCTYPE html>

<html lang="en">

<head>

  {% block meta %}

  {% endblock %}

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <link rel="icon" href="/static/img/NicoLabs-FAVICON.ico" type="image/x-icon" />

  <link rel="preconnect" href="https://fonts.googleapis.com">

  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

  <link href="https://fonts.googleapis.com/css2?family=Poppins:ital,wght@0,100;0,200;0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,300;1,400;1,500;1,600;1,700;1,800;1,900&display=swap" rel="stylesheet">

  <link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">

  <link rel="stylesheet" href="/static/css/style.css">

  <script defer src="/static/js/index.js"></script>

  <title>NicoLabs ®</title>

</head>

<body>

  <div class="container">

  {% block body %}

  {% endblock %}

  </div>

</body>

</html>

## 2.2 Página Inicial

Interface gráfica do usuário, Site

Descrição gerada automaticamente

## 2.3 Dados Históricos

Interface gráfica do usuário

Descrição gerada automaticamente com confiança baixa

## 2.4 Flask Login e Cadastro

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

## 2.5 Cadastro e listagem sensores e atuadores

Interface gráfica do usuário, Aplicativo, Teams

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

# 3 Banco de Dados

## 3.1 Modelos SQLAlchemy

\_\_init\_\_.py:

from models.userModel import Users

from models.sensorModel import Sensor

from models.atuadorModel import Atuador

from models.dadosModel import Dados

from models.leituraModel import Leitura

atuadorModel.py:

from database.config import db

class Atuador(db.Model):

  \_\_tablename\_\_ = 'atuadores'

  id = db.Column('id', db.Integer, primary\_key=True)

  apelido = db.Column(db.String(50))

  ativo = db.Column(db.Boolean)

  tipo = db.Column(db.String(50))

dadosModel.py:

Dados = {

  "temperatura": 0,

  "umidade": 0,

  "gas": 0

}

leituraModel.py:

from database.config import db

from datetime import datetime

class Leitura(db.Model):

  \_\_tablename\_\_ = 'leituras'

  id = db.Column('id', db.Integer, primary\_key=True)

  sensor = db.Column(db.String(50))

  medicao = db.Column(db.String(50))

  data = db.Column(db.DateTime, default=datetime.now)

sensorModel.py:

from database.config import db

class Sensor(db.Model):

  \_\_tablename\_\_ = 'sensores'

  id = db.Column('id', db.Integer, primary\_key=True)

  apelido = db.Column(db.String(50))

  medicao = db.Column(db.String(5))

  ativo = db.Column(db.Boolean)

userModel.py:

from database.config import db

class Users(db.Model):

  \_\_tablename\_\_ = 'users'

  id = db.Column('id', db.Integer, primary\_key=True)

  name = db.Column(db.String(50))

  email = db.Column(db.String(50))

  senha = db.Column(db.String(50))

  admin = db.Column(db.Boolean)

  operador = db.Column(db.Boolean)

## 3.2 Métodos de inserir, selecionar, editar e excluir

Os métodos estão presentes dentro do arquivo indexController.py, que pode ser encontrado nesse relatório na parte *2.1 Controladores e visualizações*. Aqui está só o código específico dos métodos:

# SENSORES ------------------------------------------------------------------------------------------------------------

@index\_bp.route('/sensores')

def sensores():

  sensores = Sensor.query.all()

  total = len(sensores)

  sensores\_ativos = Sensor.query.filter\_by(ativo=True).all()

  totalAtivos = len(sensores\_ativos)

  if session['admin'] != True and session['operador'] != True:

    return redirect(url\_for('index.index'))

  else:

    return render\_template("pages/sensores/sensores.html", sensores=sensores, total=total, totalAtivos=totalAtivos)

@index\_bp.route('/criaSensor', methods=['POST'])

def criarSensor():

  nome = request.form['nomeAddSensor']

  medicao = request.form['medicaoAddSensor']

  ativo = request.form['ativoAddSensor'] == 'True'

  novo\_sensor = Sensor(apelido=nome, medicao=medicao, ativo=ativo)

  db.session.add(novo\_sensor)

  db.session.commit()

  return redirect(url\_for('index.sensores'))

@index\_bp.route('/editarSensor', methods=['POST'])

def editarSensor():

  nome = request.form['nomeEditarSensor']

  medicao = request.form['medicaoEditarSensor']

  ativo = request.form['ativoEditarSensor'] == 'True'

  \_idSensor = request.form['idEditarSensor']

  sensor = Sensor.query.get(\_idSensor)

  if sensor:

      sensor.apelido = nome

      sensor.medicao = medicao

      sensor.ativo = ativo

      db.session.commit()

  return redirect(url\_for('index.sensores'))

@index\_bp.route('/removerSensor/<\_id>')

def removerSensor(\_id):

  sensor = Sensor.query.get(\_id)

  if sensor:

      db.session.delete(sensor)

      db.session.commit()

  return redirect(url\_for('index.sensores'))

# ATUADORES --------------------------------------------------------------------------------------------------------------

@index\_bp.route('/atuadores')

def atuadores():

  atuadores = Atuador.query.all()

  total = len(atuadores)

  atuadores\_ativos = Atuador.query.filter\_by(ativo=True).all()

  totalAtivos = len(atuadores\_ativos)

  if session['admin'] != True and session['operador'] != True:

    return redirect(url\_for('index.index'))

  else:

    return render\_template("pages/atuadores/atuadores.html", atuadores=atuadores, total=total, totalAtivos=totalAtivos)

@index\_bp.route('/criaAtuador', methods=['POST'])

def criarAtuador():

  nome = request.form['nomeAddAtuador']

  ativo = request.form['ativoAddAtuador'] == 'True'

  tipo = request.form['tipoAddAtuador']

  novo\_atuador = Atuador(apelido=nome, ativo=ativo, tipo=tipo)

  db.session.add(novo\_atuador)

  db.session.commit()

  return redirect(url\_for('index.atuadores'))

@index\_bp.route('/editarAtuador', methods=['POST'])

def editarAtuador():

  nome = request.form['nomeEditarAtuador']

  ativo = request.form['ativoEditarAtuador'] == 'True'

  \_idAtuador = request.form['idEditarAtuador']

  atuador = Atuador.query.get(\_idAtuador)

  if atuador:

      atuador.apelido = nome

      atuador.ativo = ativo

      db.session.commit()

  return redirect(url\_for('index.atuadores'))

@index\_bp.route('/removerAtuador/<\_id>')

def removerAtuador(\_id):

  atuador = Atuador.query.get(\_id)

  if atuador:

    db.session.delete(atuador)

    db.session.commit()

  return redirect(url\_for('index.atuadores'))

# USUÁRIOS --------------------------------------------------------------------------------------------------------------

@index\_bp.route('/usuarios')

def usuarios():

  users = Users.query.all()

  total = len(users)

  users\_admins = Users.query.filter\_by(admin=True).all()

  totalAdmins = len(users\_admins)

  if session['admin'] != True:

    return redirect(url\_for('index.index'))

  else:

    return render\_template("pages/usuarios/usuarios.html", users=users, total=total, totalAdmins=totalAdmins)

# LEITURAS --------------------------------------------------------------------------------------------------------------

@index\_bp.route('/leituras')

def leituras():

  leituras = Leitura.query.all()

  # Formata a data para exibição

  for leitura in leituras:

      leitura.data = leitura.data.strftime("%H:%M %d/%m/%Y")

  if session['admin'] != True:

    return redirect(url\_for('index.index'))

  else:

    return render\_template("pages/leituras/leituras.html", leituras=leituras)

# 4 Hardware

## 4.1 Esquemático do Hardware

Diagrama

Descrição gerada automaticamente

## 4.2 Tópicos MQTT utilizados

Foram utilizados os seguintes tópicos:

- bflnr/valorUmidade: Tópico para o envio dos valores de umidade, lidos pelo DHT22;

- bflnr/valorTemperatura: Tópico para o envio dos valores de temperatura, lidos pelo DHT22;

- bflnr/valorGas: Tópico para o envio dos valores simulados de gás, como não há um sensor de gás no Woki, utilizamos um sensor DHT22 para simular um.

# 5 Integração

## 5.1 Integração Web, Banco de Dados e Hardare

Uma imagem contendo Diagrama

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Tabela

Descrição gerada automaticamente

## 5.2 Comunicação IoT Flask-Mqtt

Código appController.py que realiza a conexão com o MQTT:

from flask import Flask

from flask\_mqtt import Mqtt

from database.config import db, uri

from models import Leitura

from models.dadosModel import Dados

# Inicialização do MQTT

mqtt\_client = Mqtt()

# Lista de tópicos para inscrição

topicos\_inscritos = [

    "bflnr/valorUmidade",

    "bflnr/valorTemperatura",

    "bflnr/valorGas",

]

def create\_app():

    app = Flask(

        \_\_name\_\_,

        template\_folder="./views/",

        static\_folder="./static/",

        root\_path="./"

    )

    app.config['SECRET\_KEY'] = 'secret-key'

    app.config["SQLALCHEMY\_DATABASE\_URI"] = uri

    db.init\_app(app)

    # Configurações do MQTT

    app.config['MQTT\_BROKER\_URL'] = 'mqtt-dashboard.com'

    app.config['MQTT\_BROKER\_PORT'] = 1883

    app.config['MQTT\_KEEPALIVE'] = 5000  # KeepAlive time in seconds

    # Inicializa o MQTT no contexto da aplicação

    mqtt\_client.init\_app(app)

    # Registra os blueprints

    from controllers.indexController import index\_bp

    app.register\_blueprint(index\_bp, url\_prefix='/')

    from controllers.loginController import login\_bp

    app.register\_blueprint(login\_bp, url\_prefix='/auth')

    # Define os callbacks do MQTT

    define\_mqtt\_callbacks(app)

    return app

def define\_mqtt\_callbacks(app):

    @mqtt\_client.on\_connect()

    def handle\_connect(client, userdata, flags, rc):

        if rc == 0:

            print('Broker Conectado com sucesso')

            for topic in topicos\_inscritos:

                mqtt\_client.subscribe(topic)  # Se inscreve nos tópicos

        else:

            print('Bad connection. Code:', rc)

    @mqtt\_client.on\_disconnect()

    def handle\_disconnect(client, userdata, rc):

        print("Broker desconectado")

    @mqtt\_client.on\_message()

    def handle\_mqtt\_message(client, userdata, message):

        with app.app\_context():

            payload = message.payload.decode("utf-8")  # Decodifica o payload como string

            topic = str(message.topic)

            sensor = ""

            print("\*\*\*\*\*\*\* ", topic, payload)

            if topic == "bflnr/valorUmidade":

                Dados['umidade'] = float(payload)

                sensor = "Umidade"

            elif topic == "bflnr/valorTemperatura":

                Dados['temperatura'] = float(payload)

                sensor = "Temperatura"

            elif topic == "bflnr/valorGas":

                Dados['gas'] = float(payload)

                sensor = "Gás"

            else:

                return  # Ignora tópicos não reconhecidos

            nova\_leitura = Leitura(sensor=sensor, medicao=payload)

            db.session.add(nova\_leitura)

            db.session.commit()

def send\_mqtt\_message(topic, message):

    mqtt\_client.publish(topic, message)

Parte do código indexController.py que realiza o acionamento do LED e do Buzzer no Hardware e que joga as leituras para o banco de dados:

@index\_bp.route('/leituras')

def leituras():

  leituras = Leitura.query.all()

  # Formata a data para exibição

  for leitura in leituras:

      leitura.data = leitura.data.strftime("%H:%M %d/%m/%Y")

  if session['admin'] != True:

    return redirect(url\_for('index.index'))

  else:

    return render\_template("pages/leituras/leituras.html", leituras=leituras)

@index\_bp.route('/acionaLed')

def acionaLed():

  valor = '0'

  atuador = Atuador.query.filter\_by(apelido="LED").first()

  atuador.ativo = atuador.ativo != True # INVERTE O VALOR DO SENSOR

  db.session.commit()

  if atuador.ativo:

    valor = '1'

  else:

    valor = '0'

  send\_mqtt\_message('bflnr/valorLed', valor)

  return redirect(url\_for('index.index'))

@index\_bp.route('/acionaBuzzer')

def acionaBuzzer():

  send\_mqtt\_message('bflnr/valorBuzzer', '1')

  return '', 204

Leituras no banco de dados:

Interface gráfica do usuário, Tabela

Descrição gerada automaticamente com confiança média