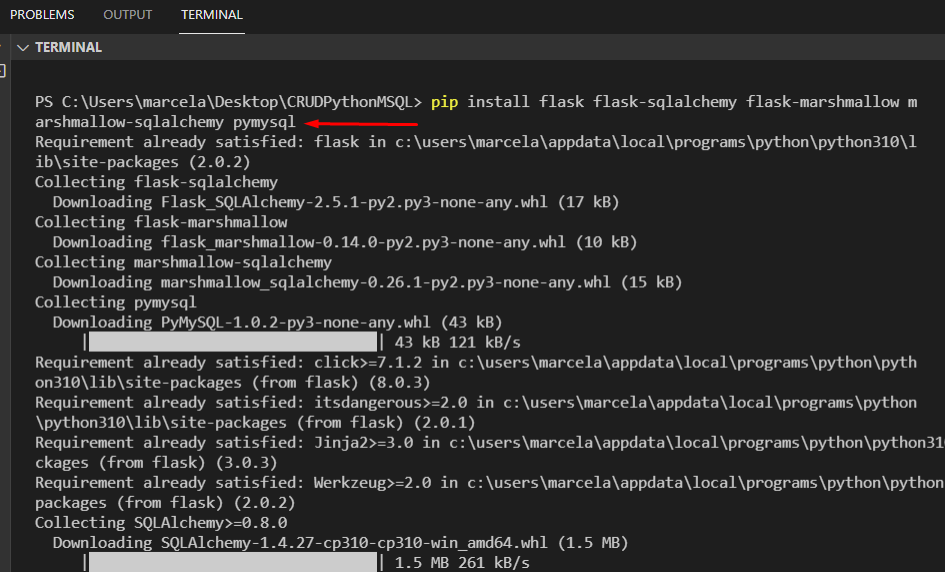
Vamos a crear una carpeta en el escritorio que se llame por ejemplo: CRUDPythonMySQL

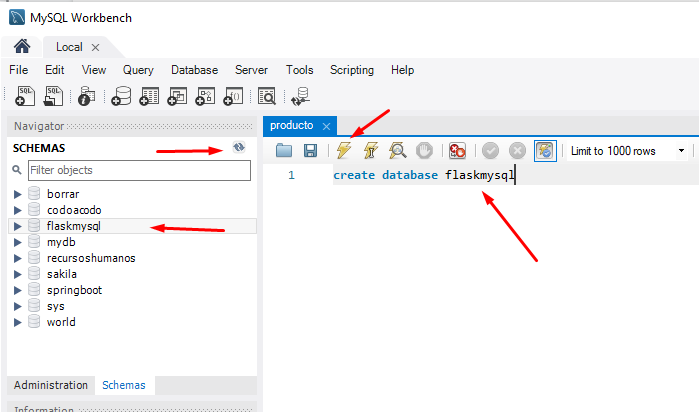
Desde VSC abrimos la carpeta que acabamos de crear

Instalar en la terminal: pip install flask flask-sqlalchemy flask-marshmallow marshmallow-sqlalchemy pymysql



En la terminal instalar también : pip install -U flask-cors (me va a permitir acceder desde el frontend al Json que genera el backend)

Crear una base de datos que se llame flaskmysql, refrescar y ver que se haya creado



Crear una carpeta src, dentro de ella, crear un archivo app.py

from flask import Flask ,jsonify,request

from flask\_cors import CORS

from flask\_sqlalchemy import SQLAlchemy

from flask\_marshmallow import Marshmallow

app=Flask(\_\_name\_\_)

CORS(app)

# configuro la base de datos, con el nombre el usuario y la clave

app.config['SQLALCHEMY\_DATABASE\_URI']='mysql+pymysql://root:root@localhost/flaskmysql'

# user:clave@localhost/nombreBaseDatos

app.config['SQLALCHEMY\_TRACK\_MODIFICATIONS']=False

db= SQLAlchemy(app)

ma=Marshmallow(app)

# defino la tabla

class Producto(db.Model): # la clase Producto hereda de db.Model

id=db.Column(db.Integer, primary\_key=True) #define los campos de la tabla

nombre=db.Column(db.String(100))

precio=db.Column(db.Integer)

stock=db.Column(db.Integer)

def \_\_init\_\_(self,nombre,precio,stock): #crea el constructor de la clase

self.nombre=nombre # no hace falta el id porque lo crea sola mysql por ser auto\_incremento

self.precio=precio

self.stock=stock

db.create\_all() # crea las tablas

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

class ProductoSchema(ma.Schema):

class Meta:

fields=('id','nombre','precio','stock')

producto\_schema=ProductoSchema() # para crear un producto

productos\_schema=ProductoSchema(many=True) # multiples registros

@app.route('/productos',methods=['GET'])

def get\_Productos():

all\_productos=Producto.query.all()

result=productos\_schema.dump(all\_productos)

return jsonify(result)

@app.route('/productos/<id>',methods=['GET'])

def get\_producto(id):

producto=Producto.query.get(id)

return producto\_schema.jsonify(producto)

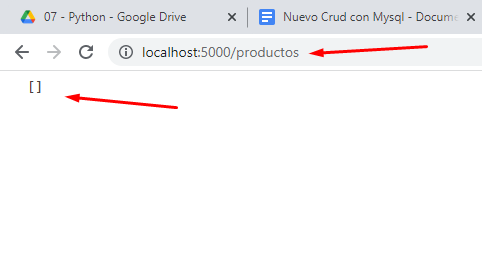
# programa principal

if \_\_name\_\_=='\_\_main\_\_':

app.run(debug=True, port=4000)

y correr en la terminal: python src\app.py

probar en el navegador <http://localhost:5000/productos> , si devuelve [ ] es que esta’ correcto, el json que trajo los datos de la base de datos está vacío



cargar en la base de datos 2 o 3 productos y volver a probar en el navegador <http://localhost:5000/productos> y tiene que mostrar el json con los datos que acaban de crear

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* hasta aca comision 2172 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Crear un frontend que contenga una tabla para mostrar los productos



Crear un archivo index.html

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Document</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p"

crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>

</head>

<body>

<main>

<div class="container" id="app">

<h1>Productos</h1>

<section v-if="errored">

<p>Lo sentimos, no es posible obtener la información en este momento, por favor intente nuevamente mas tarde</p>

</section>

<section v-else>

<div v-if="loading">Cargando....

<div class="spinner-border" role="status">

<span class="sr-only"></span>

</div>

</div>

<div v-else>

<a href="create.html" class="btn btn-success">Nuevo</a>

<br><br>

<table class="table table-striped table-responsive">

<tr>

<th>Id</th>

<th>Nombre</th>

<th>Precio</th>

<th>Stock</th>

<th>Acciones</th>

</tr>

<tr v-for="producto in productos">

<td>{{producto.id}}</td>

<td>{{producto.nombre}}</td>

<td>{{producto.precio}}</td>

<td>{{producto.stock}}</td>

<td>

<a :href="'update.html?id='+producto.id+'&nombre='+producto.nombre+'&precio='+producto.precio+'&stock='+producto.stock" class="btn btn-primary">Modificar</a>

<button v-on:click="eliminar(producto.id)" class="btn btn-danger">Eliminar</button>

</td>

</tr>

</table>

</section>

</div>

</main>

<script src="./js/main.js"></script>

</body>

</html>

Crea una carpeta JS y un archivo main.js

if (document.getElementById("app")) {

const app = new Vue({

el: "#app",

data: {

productos: [],

errored: false,

loading: true

},

created() {

var url = 'http://localhost:5000/productos'

this.fetchData(url)

},

methods: {

fetchData(url) {

fetch(url)

.then(response => response.json())

.then(data => {

this.productos = data;

this.loading = false;

})

.catch(err => {

this.errored = true

})

},

eliminar(producto) {

const url = 'http://localhost:5000/producto/' + producto;

var options = {

method: 'DELETE',

}

fetch(url, options)

.then(res => res.text()) // or res.json()

.then(res => {

location.reload();

})

}

}

})

}

Agregar la ruta o endpoint /productos para el método POST

@app.route('/productos', methods=['POST']) # crea ruta o endpoint

def create\_producto():

print(request.json) # request.json contiene el json que envio el cliente

nombre=request.json['nombre']

precio=request.json['precio']

stock=request.json['stock']

new\_producto=Producto(nombre,precio,stock)

db.session.add(new\_producto)

db.session.commit()

return producto\_schema.jsonify(new\_producto)

crear un archivo create.html

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Document</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p" crossorigin="nonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>

</head>

<body>

<main style="margin: 1% 20%;">

<div class="container">

<h1>Alta Producto</h1>

<form>

<div class="mb-3">

<label for="txtNombre">Nombre</label>

<input type="text" class="form-control" name="txtNombre" id="txtNombre"><br>

</div>

<div class="mb-3">

<label for="txtPrecio">Precio</label>

<input type="text" class="form-control" name="txtPrecio" id="txtPrecio"> <br>

</div>

<div class="mb-3">

<label for="txtStock">Stock</label>

<input type="number" class="form-control" name="txtStock" id="txtStock"> <br>

</div>

<button type="button" class="btn btn-success" onclick="guardar()">Grabar</button>

</form>

<br>

<a href="index.html" class="btn btn-warning">Home</a>

</div>

</main>

<script src="./js/create.js"></script>

</body>

</html>

crear un archivo create.js

function guardar() {

let n = document.getElementById("txtNombre").value

let p = parseFloat(document.getElementById("txtPrecio").value)

let s = parseInt(document.getElementById("txtStock").value)

let producto = {

nombre: n,

precio: p,

stock: s

}

let url = "http://localhost:5000/productos"

var options = {

body: JSON.stringify(producto),

method: 'POST',

headers: { 'Content-Type': 'application/json' },

// redirect: 'follow'

}

fetch(url, options)

.then(function () {

console.log("creado")

alert("Grabado")

// Handle response we get from the API

})

.catch(err => {

//this.errored = true

alert("Error al grabar" )

console.error(err);

})

}

Ya se puede probar crear un nuevo producto

Agregar la ruta o endpoint /productos/id para el método PUT

@app.route('/productos/<id>' ,methods=['PUT'])

def update\_producto(id):

producto=Producto.query.get(id)

nombre=request.json['nombre']

precio=request.json['precio']

stock=request.json['stock']

producto.nombre=nombre

producto.precio=precio

producto.stock=stock

db.session.commit()

return producto\_schema.jsonify(producto)

Crear un archivo update.html

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Document</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p" crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>

</head>

<body>

<main style="margin: 1% 20%;">

<div class="container">

<h1>Modificacion Productos</h1>

<form >

<div class="mb-3">

<label for="txtId">Id</label>

<input type="text" class="form-control" name="txtId" id="txtId" disabled ><br>

</div>

<div class="mb-3">

<label for="txtNombre">Nombre:</label>

<input type="text" class="form-control" name="txtNombre" id="txtNombre"><br>

</div>

<div class="mb-3">

<label for="txtPrecio">Precio:</label>

<input type="text" class="form-control" name="txtPrecio" id="txtPrecio"> <br>

</div>

<div class="mb-3">

<label for="txtStock">Stock:</label>

<input type="number" class="form-control" name="txtStock" id="txtStock"> <br>

</div>

<button type="button" class="btn btn-success" onclick="modificar()">Grabar</button>

</form>

<br>

<a href="index.html" class="btn btn-warning">Home</a>

</div>

</main>

<script src="./js/update.js"></script>

</body>

</html>

y un archivo update.js

var args = location.search.substr(1).split('&');

// lee los argumentos pasados a este formulario

var parts = []

for (let i = 0; i < args.length; ++i) {

parts[i] = args[i].split('=');

}

console.log(args)

document.getElementById("txtId").value = parts[0][1]

document.getElementById("txtNombre").value = parts[1][1]

document.getElementById("txtPrecio").value = parts[2][1]

document.getElementById("txtStock").value = parts[3][1]

function modificar() {

let id = document.getElementById("txtId").value

let n = document.getElementById("txtNombre").value

let p = parseFloat(document.getElementById("txtPrecio").value)

let s = parseInt(document.getElementById("txtStock").value)

let producto = {

nombre: n,

precio: p,

stock: s

}

let url = "http://localhost:5000/productos/"+id

var options = {

body: JSON.stringify(producto),

method: 'PUT',

headers: { 'Content-Type': 'application/json' },

redirect: 'follow'

}

fetch(url, options)

.then(function () {

console.log("modificado")

alert("Registro modificado")

// Handle response we get from the API

})

.catch(err => {

//this.errored = true

console.error(err);

alert("Error al Modificar")

})

}

Ya se puede probar modificar un registro

Tiene un error, cuando se ejecuta paso a paso funciona bien, pero si se ejecuta todo de una vez, no graba

Agregar la ruta o endpoint /productos/id para el método DELETE

@app.route('/producto/<id>',methods=['DELETE'])

def delete\_producto(id):

producto=Producto.query.get(id)

db.session.delete(producto)

db.session.commit()

return producto\_schema.jsonify(producto)

ya se puede probar el botón eliminar