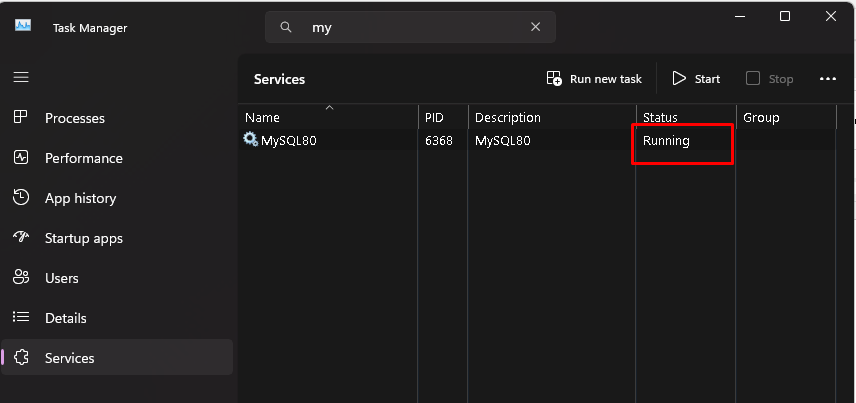
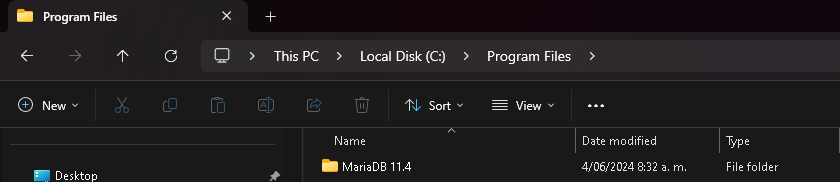
# GUIA PARA LEVANTAR SOFTWARE INVENTARIOS

# 1.Tener instalado MariaDB y arriba el servicio de MySQL Server

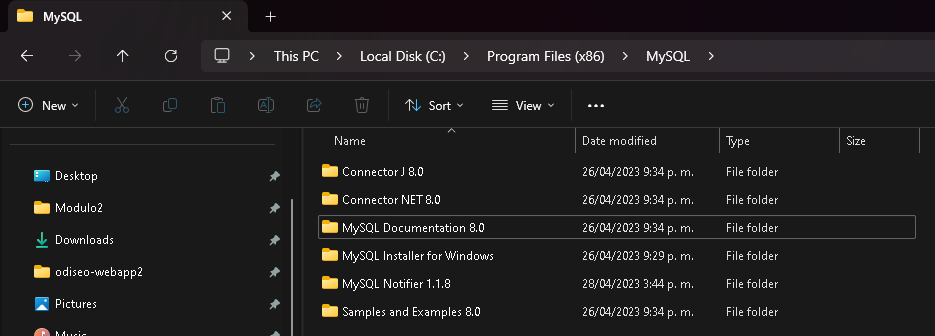
Services.msc



C:\Program Files

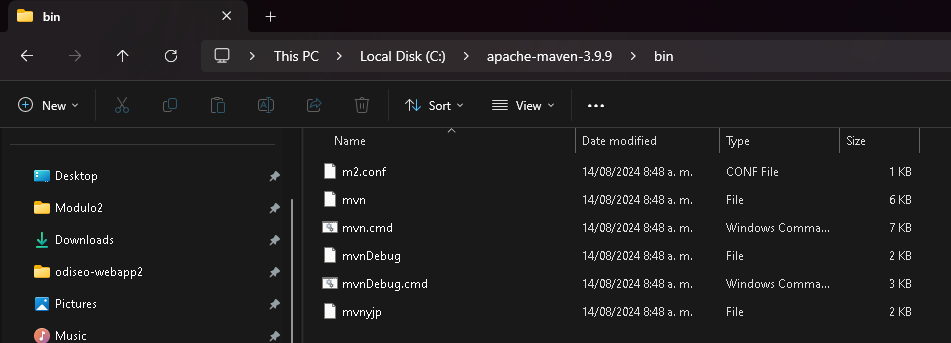


C:\Program Files (x86)\MySQL



# 2.Tener instalado el Maven (Para proyectos de Spring Boot)

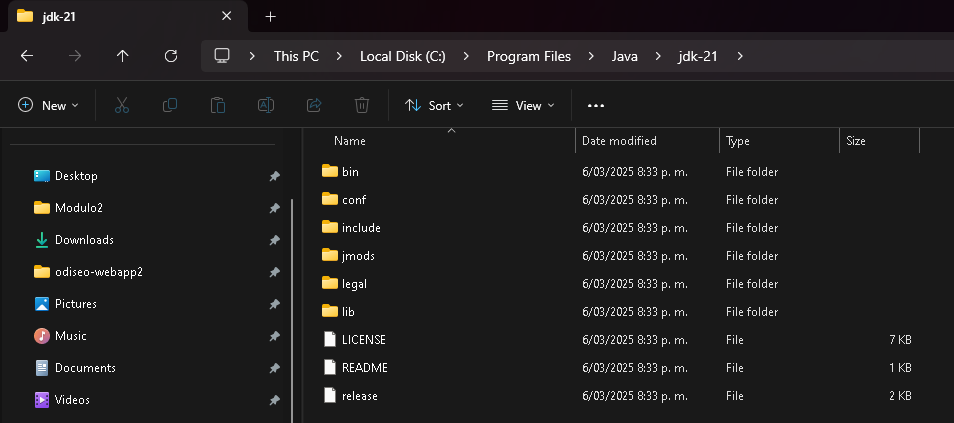
C:\apache-maven-3.9.9\bin



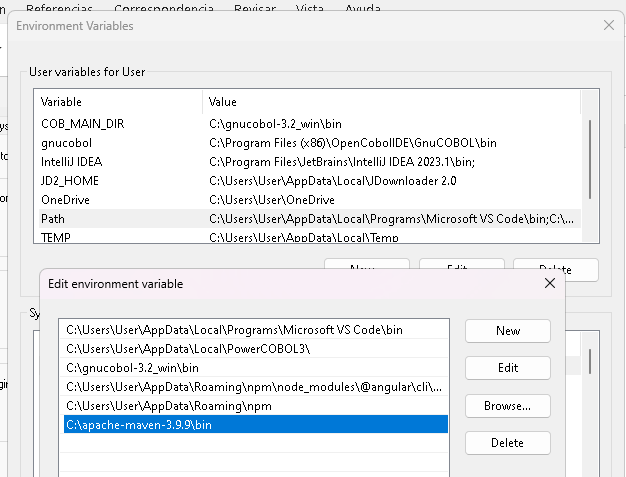
# 3.Tener instalado el Java con la versión que se vaya a aplicar (en este caso la versión 21)

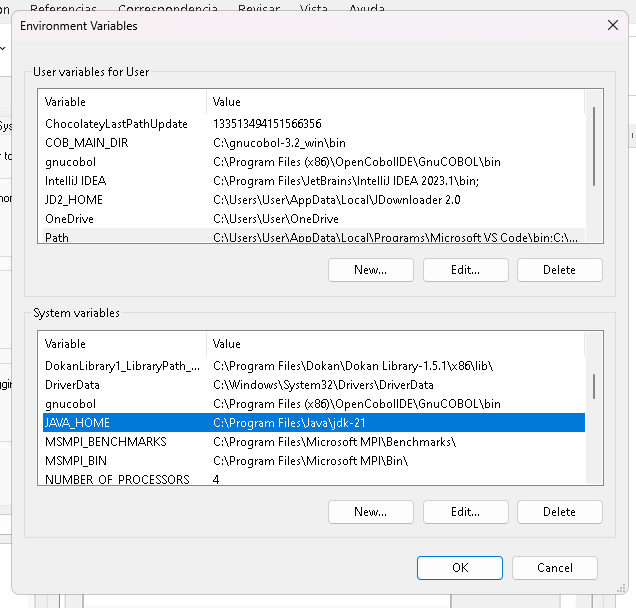


C:\Program Files\Java\jdk-21\bin



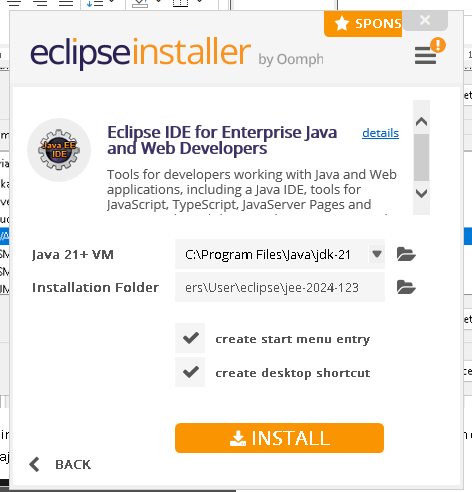
# 4.Tener en las variables de entorno configurados el path del java y del Maven





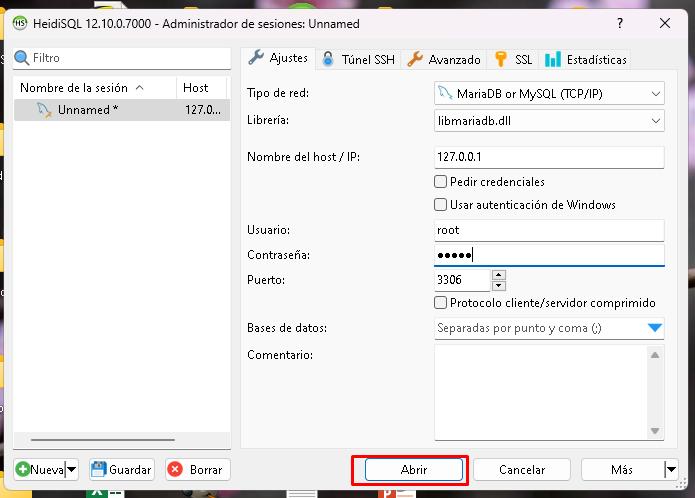
# 5. Tener instalado un IDE para trabajar con Java, en este caso Eclipse, preferiblemente uno que trabaje con la versión de Java que se instaló.

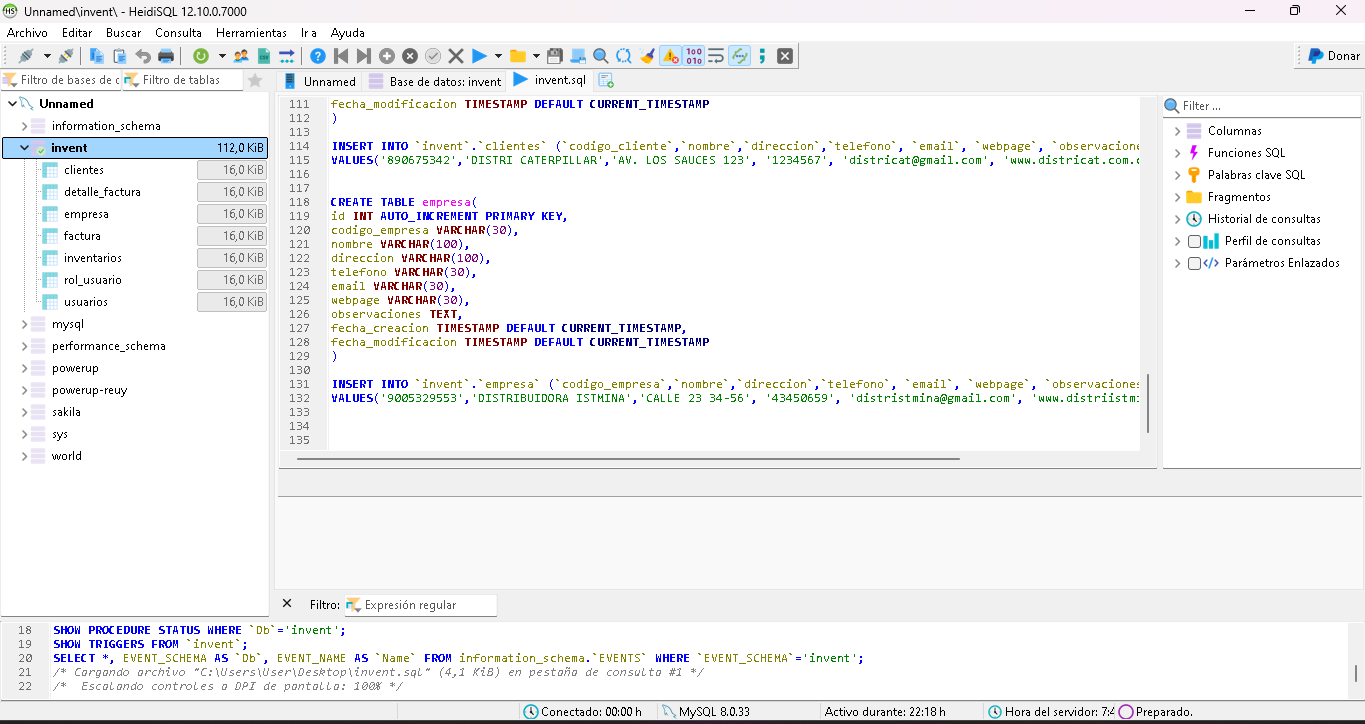




# 6.Se crea y configura una base de datos en MariaDB. Para este caso se utiliza HeidySQL como IDE para gestionar la base de datos.







# Base de datos de prueba.

inventariosinventariosUSE invent;

**DROP** **TABLE** usuarios;

**CREATE** **TABLE** usuarios(

id **INT** **AUTO\_INCREMENT** **PRIMARY** **KEY**,

nombre **VARCHAR**(100) **NOT** **NULL**,

contrasena **VARCHAR**(255) **NOT** **NULL**,

codrol **INT** **NOT** **NULL** **DEFAULT** 999,

fecha\_creacion **TIMESTAMP** **DEFAULT** **CURRENT\_TIMESTAMP**,

fecha\_actualizacion **TIMESTAMP** **DEFAULT** **CURRENT\_TIMESTAMP** **ON** **UPDATE** **CURRENT\_TIMESTAMP**

);

**INSERT** **INTO** invent.usuarios(nombre, contrasena) **VALUES**

('SYSADMIN',**SHA2**('SYSADMIN',256));

**INSERT** **INTO** invent.usuarios(nombre, contrasena) **VALUES**

('ADMEMP',**SHA2**('ADMEMP',256));

**INSERT** **INTO** invent.usuarios(nombre, contrasena) **VALUES**

('ADMEMP',**SHA2**('ADMEMP',256));

**INSERT** **INTO** invent.usuarios(nombre, contrasena) **VALUES**

('FUNC1',**SHA2**('FUNC1',256));

**CREATE** **TABLE** rol\_usuario(

codrol **INT** **NOT** **NULL**,

descrol **VARCHAR**(50),

observaciones **VARCHAR**(200) **DEFAULT** 'Ninguna'

)

**INSERT** **INTO** invent.rol\_usuario(codrol, descrol) **VALUES**

(1,'Superusuario');

**INSERT** **INTO** invent.rol\_usuario(codrol, descrol) **VALUES**

(2,'Administradores');

**INSERT** **INTO** invent.rol\_usuario(codrol, descrol) **VALUES**

(3,'Gerente');

**INSERT** **INTO** invent.rol\_usuario(codrol, descrol) **VALUES**

(4,'Supervisores');

**INSERT** **INTO** invent.rol\_usuario(codrol, descrol) **VALUES**

(5,'Funcionarios');

**INSERT** **INTO** invent.rol\_usuario(codrol, descrol) **VALUES**

(999,'Rol por defecto');

**SELECT** \* **FROM** usuarios;

**CREATE** **TABLE** inventarios(

id **INT** **AUTO\_INCREMENT** **PRIMARY** **KEY**,

codigo **VARCHAR**(50) **NOT** **NULL**,

nombre\_producto **VARCHAR**(100) **NOT** **NULL**,

unidad\_medida **VARCHAR**(30),

cantidad **INT** **NOT** **NULL**,

precio **DECIMAL**(10,2) **NOT** **NULL**,

observaciones **TEXT**,

fecha\_creacion **TIMESTAMP** **DEFAULT** **CURRENT\_TIMESTAMP**,

fecha\_modificacion **TIMESTAMP** **DEFAULT** **CURRENT\_TIMESTAMP**

);

**INSERT** **INTO** `invent`.`inventarios` (`codigo`, `nombre\_producto`, `unidad\_medida`, `cantidad`, `precio`, `observaciones`)

**VALUES** ('ABCD12345', 'Caucho transmisión', 'unidad', 100, 12000, 'Pendiente inventariar');

**INSERT** **INTO** `invent`.`inventarios` (`codigo`, `nombre\_producto`, `unidad\_medida`, `cantidad`, `precio`, `observaciones`)

**VALUES** ('EFGHI67890', 'Bomba hidráulica', 'unidad', 10, 150000, 'Pendiente inventariar');

**SELECT** \* **FROM** invent.inventarios;

**DROP** **TABLE** factura;

**CREATE** **TABLE** factura(

id **INT** **AUTO\_INCREMENT** **PRIMARY** **KEY**,

codigo **VARCHAR**(50),

id\_cliente **VARCHAR**(30) **NOT** **NULL**,

fecha\_factura **TIMESTAMP** **DEFAULT** **CURRENT\_TIMESTAMP**,

total **DECIMAL**(10,2) **NOT** **NULL**,

estado\_pago **VARCHAR**(20) **NOT** **NULL**,

fecha\_creacion **TIMESTAMP** **DEFAULT** **CURRENT\_TIMESTAMP**,

fecha\_modificacion **TIMESTAMP** **DEFAULT** **CURRENT\_TIMESTAMP**

)

**CREATE** **TABLE** detalle\_factura(

codigo\_factura **VARCHAR**(50),

codigo\_producto **VARCHAR**(50),

cantidad **INT**

)

**SELECT** \* **FROM** factura;

**INSERT** **INTO** `invent`.`factura` (`codigo`,`id\_cliente`,`total`,`estado\_pago`)

**VALUES**('0001-0034','890675342',12000,'PENDIENTE');

**INSERT** **INTO** `invent`.`detalle\_factura` (`codigo\_factura`,`codigo\_producto`,`cantidad`)

**VALUES**('0001-0034','ABCD12345',1);

**CREATE** **TABLE** clientes(

id **INT** **AUTO\_INCREMENT** **PRIMARY** **KEY**,

codigo\_cliente **VARCHAR**(30),

nombre **VARCHAR**(100),

direccion **VARCHAR**(100),

telefono **VARCHAR**(30),

email **VARCHAR**(30),

webpage **VARCHAR**(30),

observaciones **TEXT**,

fecha\_creacion **TIMESTAMP** **DEFAULT** **CURRENT\_TIMESTAMP**,

fecha\_modificacion **TIMESTAMP** **DEFAULT** **CURRENT\_TIMESTAMP**

)

**INSERT** **INTO** `invent`.`clientes` (`codigo\_cliente`,`nombre`,`direccion`,`telefono`, `email`, `webpage`, `observaciones`)

**VALUES**('890675342','DISTRI CATERPILLAR','AV. LOS SAUCES 123', '1234567', 'districat@gmail.com', 'www.districat.com.co', 'cliente prioritario');

**CREATE** **TABLE** empresa(

id **INT** **AUTO\_INCREMENT** **PRIMARY** **KEY**,

codigo\_empresa **VARCHAR**(30),

nombre **VARCHAR**(100),

direccion **VARCHAR**(100),

telefono **VARCHAR**(30),

email **VARCHAR**(30),

webpage **VARCHAR**(30),

observaciones **TEXT**,

fecha\_creacion **TIMESTAMP** **DEFAULT** **CURRENT\_TIMESTAMP**,

fecha\_modificacion **TIMESTAMP** **DEFAULT** **CURRENT\_TIMESTAMP**

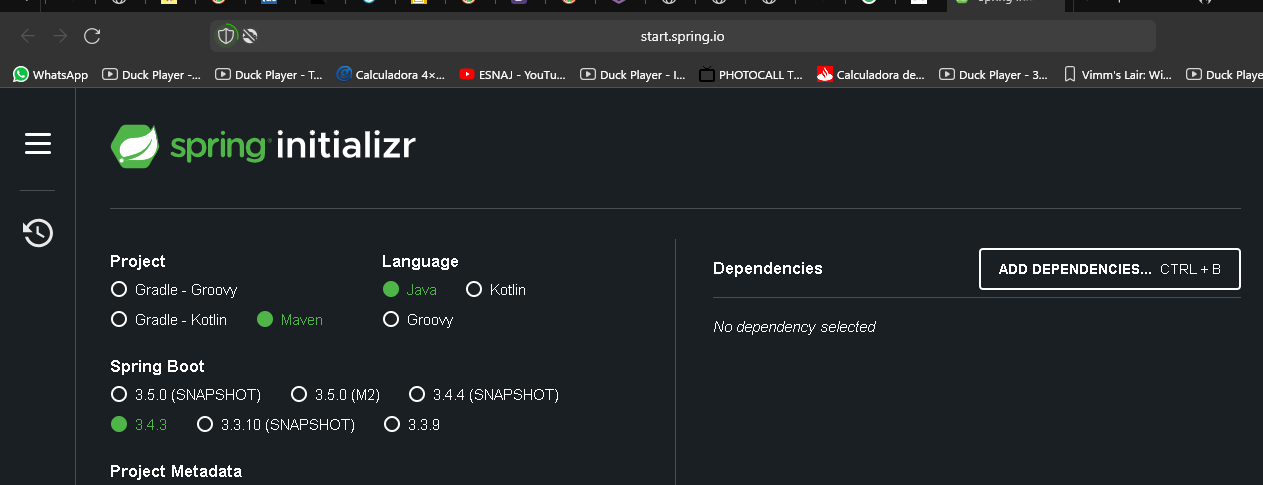
)

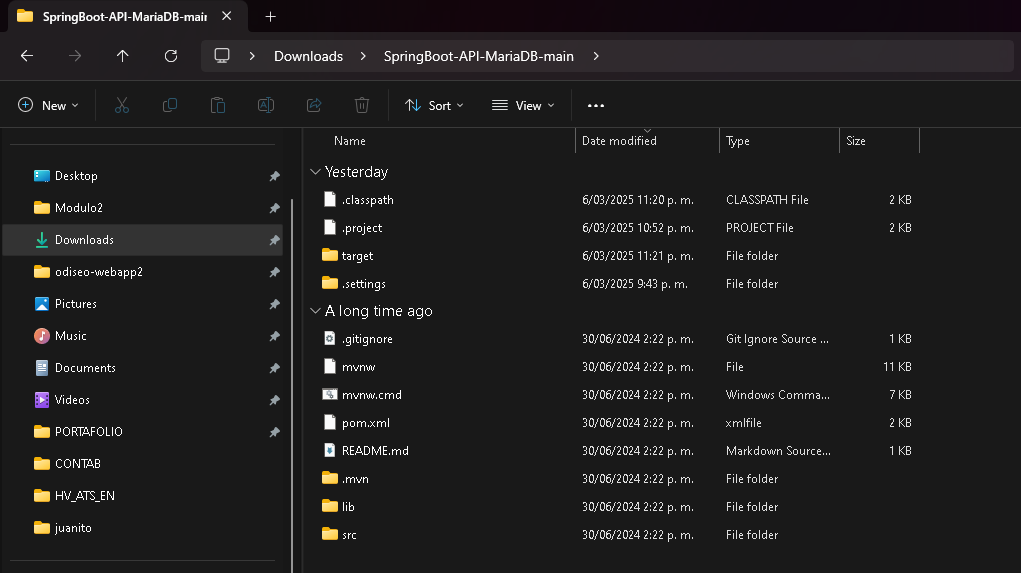
**INSERT** **INTO** `invent`.`empresa` (`codigo\_empresa`,`nombre`,`direccion`,`telefono`, `email`, `webpage`, `observaciones`)

**VALUES**('9005329553','DISTRIBUIDORA ISTMINA','CALLE 23 34-56', '43450659', 'distristmina@gmail.com', 'www.distriistmina.com.co', 'principal');

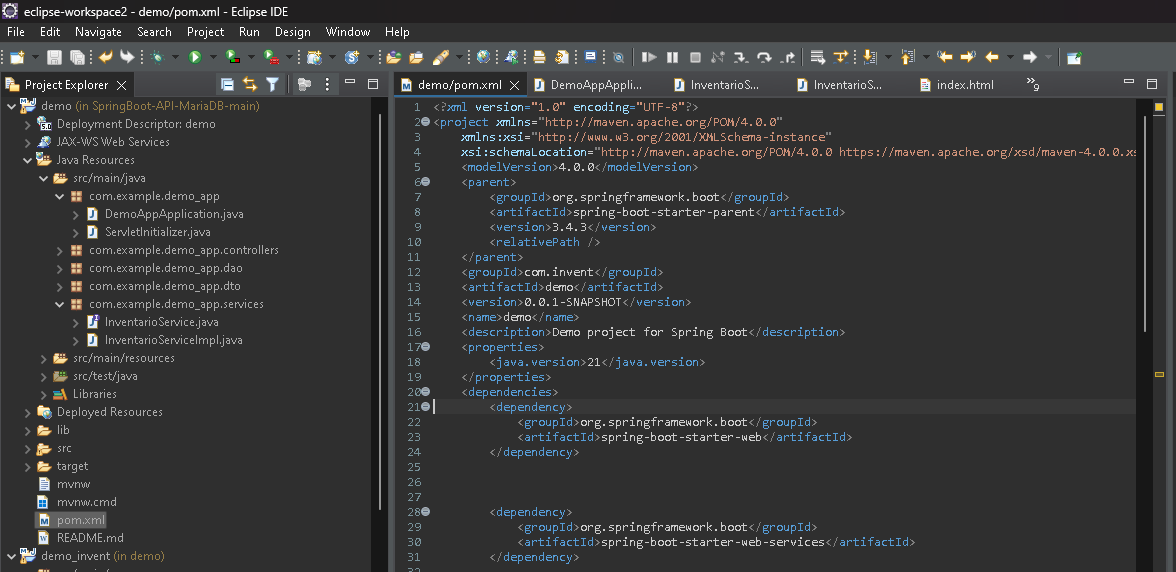
# 7.Una vez se tenga instaladas las herramientas se procede a descargar un proyecto previo de Spring Boot que tenga configurada la versión de Java y Maven como gestor de dependencias por defecto. También se puede descargar una versión con el inicializador del spring.

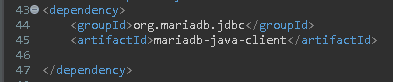
Página: https://start.spring.io/





# 8.Se configura el pom.xml con las dependencias necesarias que permitan soportar el acceso a mariadb y otras herramientas.





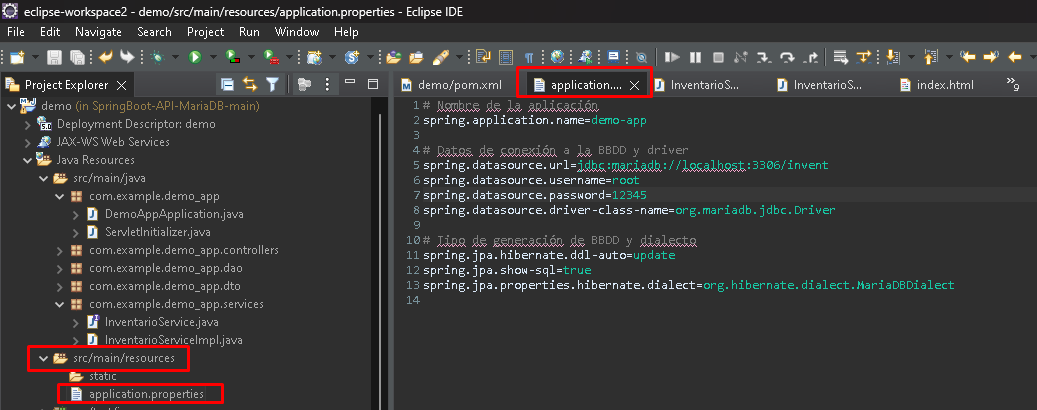
<dependency>

<groupId>org.mariadb.jdbc</groupId>

<artifactId>mariadb-java-client</artifactId>

</dependency>

# 9. Tener configurado el application.properties con los datos para acceder a la base de datos.



# Nombre de la aplicación

spring.application.name=demo-app

# Datos de conexión a la BBDD y driver

spring.datasource.url=jdbc:mariadb://localhost:3306/invent

spring.datasource.username=root

spring.datasource.password=12345

spring.datasource.driver-class-name=org.mariadb.jdbc.Driver

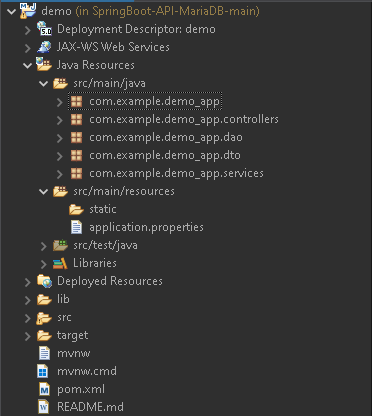
# Tipo de generación de BBDD y dialecto

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MariaDBDialect

# 10.Tener configurada la estructura de paquetes (carpetas) donde se alojaran las clases que permitirán realizar la persistencia y tratamiento de los datos.



Controller: Se implementará la lógica dentro de esta, se utilizarán anotaciones JPA y se definirán los endpoints de la aplicación.

Repository(DAO): Es una interfáz en la cual se heredarán los métodos de JPA para su posterior implementación en la capa de servicios(services).

DTO: Es el modelo de objetos a utilizar, en nuestro caso utilizaremos crearemos una clase Usuario para poder crear entidades de este tipo.

Services: Consta de una interfáz con la cabecera de los métodos la cuál se implementará en la clase de servicios(UsuarioServiceImpl) para su posterior uso en el controlador.

# 11.Configurar las clases que hacen parte de cada uno de los elementos

# Clase inventario.java paquete com.example.demo\_app.dto

package com.example.demo\_app.dto;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

@Entity

@Table(name="inventarios")

public class Inventario {

/\*@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;\*/

@Id

@Column(name = "codigo")

private String codigo;

@Column(name="nombre\_producto")

private String nombre\_producto;

@Column(name="unidad\_medida")

private String unidad\_medida;

@Column(name="cantidad")

private int cantidad ;

@Column(name="precio")

private double precio;

@Column(name="observaciones")

private String observaciones;

public Inventario() {}

public Inventario(String codigo, String nombre\_producto, String unidad\_medida, int cantidad, double precio, String observaciones) {

this.codigo = codigo;

this.nombre\_producto = nombre\_producto;

this.unidad\_medida = unidad\_medida;

this.cantidad = cantidad;

this.precio = precio;

this.observaciones = observaciones;

}

/\*public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}\*/

public String getCodigo() {

return codigo;

}

public void setCodigo(String codigo) {

this.codigo = codigo;

}

public String getNombre\_producto() {

return nombre\_producto;

}

public void setNombre\_producto(String nombre\_producto) {

this.nombre\_producto = nombre\_producto;

}

public String getUnidad\_medida() {

return unidad\_medida;

}

public void setUnidad\_medida(String unidad\_medida) {

this.unidad\_medida = unidad\_medida;

}

public int getCantidad() {

return cantidad;

}

public void setCantidad(int cantidad) {

this.cantidad = cantidad;

}

public double getPrecio() {

return precio;

}

public void setPrecio(double precio) {

this.precio = precio;

}

public String getObservaciones() {

return observaciones;

}

public void setObservaciones(String observaciones) {

this.observaciones = observaciones;

}

@Override

public String toString() {

return "Usuario [codigo=" + codigo + ", nombre=" + nombre\_producto + ", unidad\_medida=" + unidad\_medida + ", cantidad=" + cantidad + ", precio="

+ precio + ", observaciones="+ observaciones + "]";

}

}

# Clase Interface InventarioRepository paquete com.example.demo\_app.dao

package com.example.demo\_app.dao;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import com.example.demo\_app.dto.Inventario;

*@Repository*

public interface InventarioRepository extends JpaRepository<Inventario, String>{

}

# Clase Interface InventarioService.java paquete com.example.demo\_app.services

package com.example.demo\_app.services;

import java.util.List;

import java.util.Optional;

import com.example.demo\_app.dto.Inventario;

public interface InventarioService {

List<Inventario> getAllInventarios();

Optional<Inventario> getInventarioById(String codinventario);

Optional<Inventario> createInventario(Inventario inventario);

Inventario updateInventario(String codinventario, Inventario inventario);

void deleteInventarioById(String codinventario);

}

# Clase InventarioServiceImpl.java package com.example.demo\_app.services

package com.example.demo\_app.services;

import java.util.List;

import java.util.NoSuchElementException;

import java.util.Optional;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.example.demo\_app.dao.InventarioRepository;

import com.example.demo\_app.dto.Inventario;

*@Service*

public class InventarioServiceImpl implements InventarioService{

*@Autowired*

public InventarioRepository inventarioRepository;

Logger logger = LoggerFactory.*getLogger*(InventarioServiceImpl.class);

*@Override*

public List<Inventario> getAllInventarios() {

List<Inventario> inventario = inventarioRepository.findAll();

return inventario;

}

*@Override*

public Optional<Inventario> getInventarioById(String codinventario) {

Optional<Inventario> searchInventory = inventarioRepository.findById(codinventario);

if(searchInventory.isPresent()) {

logger.info("Inventario encontrado.");

return searchInventory;

} else {

logger.error("No se encontró el elemento de inventario" + codinventario + "buscado");

throw new NoSuchElementException("No se encontró el elemento de inventario" + codinventario + "buscado");

}

}

*@Override*

public Optional<Inventario> createInventario(Inventario inventario) {

Optional<Inventario> optInventario = Optional.*of*(inventario);

if(optInventario.isPresent()) {

Inventario newInventory = new Inventario();

newInventory.setCodigo(inventario.getCodigo());

newInventory.setNombre\_producto(inventario.getNombre\_producto());

newInventory.setUnidad\_medida(inventario.getUnidad\_medida());

newInventory.setCantidad(inventario.getCantidad());

newInventory.setObservaciones(inventario.getObservaciones());

inventarioRepository.save(newInventory);

logger.info("Se ha creado el elemento de inventario correctamente.");

Optional<Inventario> rturndOptInventory = Optional.*of*(newInventory);

return rturndOptInventory;

} else {

logger.error("No se ha recibido un elemento de inventario para crear.");

throw new NoSuchElementException("No se ha recibido un elemento de inventario para crear.");

}

}

*@Override*

public Inventario updateInventario(String codinventario, Inventario inventario) {

Optional<Inventario> optInventario = inventarioRepository.findById(codinventario);

if(optInventario.isPresent()) {

Inventario inventarioUpdate = optInventario.get();

inventarioUpdate.setNombre\_producto(inventario.getNombre\_producto());

inventarioUpdate.setUnidad\_medida(inventario.getUnidad\_medida());

inventarioUpdate.setCantidad(inventario.getCantidad());

inventarioUpdate.setPrecio(inventario.getPrecio());

inventarioUpdate.setObservaciones(inventario.getObservaciones());

inventarioRepository.save(inventarioUpdate);

logger.info("Inventario con id: " + codinventario + " actualizado correctamente.");

return inventarioUpdate;

} else {

logger.error("El elemento de inventario a modificar no existe");

throw new NoSuchElementException("No se encontró el elemento de inventario" + codinventario + "buscado");

}

}

*@Override*

public void deleteInventarioById(String codinventario) {

Optional<Inventario> optInventario = inventarioRepository.findById(codinventario);

if(optInventario.isPresent()) {

Inventario inventarioTrust = optInventario.get();

inventarioRepository.deleteById(inventarioTrust.getCodigo());

logger.info("El elemento de inventario con id: " + codinventario + " fue eliminado de forma efectiva.");

} else {

logger.error("El elemento de inventario a eliminar no existe.");

}

}

}

# Clase InventarioController.java package com.example.demo\_app.controllers

package com.example.demo\_app.controllers;

import java.util.List;

import java.util.NoSuchElementException;

import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.example.demo\_app.dto.Inventario;

import com.example.demo\_app.services.InventarioServiceImpl;

@RestController

@RequestMapping("/api/inventario")

@CrossOrigin(origins = "\*")

public class InventarioController {

@Autowired

public InventarioServiceImpl inventarioServiceImpl;

@GetMapping("/all")

public List<Inventario> getAllInventarios(){

return inventarioServiceImpl.getAllInventarios();

}

@GetMapping("/{id}")

public Inventario getInventarioById(@PathVariable(name="codigo") String codigo) {

Optional<Inventario> inventario = inventarioServiceImpl.getInventarioById(codigo);

if(inventario.isPresent()) {

return inventario.get();

} else {

throw new NoSuchElementException();

}

}

@PostMapping("/create")

public Optional<Inventario> createInventario(@RequestBody Inventario inventario){

Optional<Inventario> newInventario = inventarioServiceImpl.createInventario(inventario);

if(newInventario.isPresent()) {

return newInventario;

} else {

throw new NoSuchElementException();

}

}

@PutMapping("/update/{id}")

public Inventario updateInventario(@PathVariable(name="codigo") String codigo, @RequestBody Inventario inventario){

Inventario updateInventario = inventarioServiceImpl.updateInventario(codigo, inventario);

return updateInventario;

}

@DeleteMapping("/delete/{id}")

public void deleteInventario(@PathVariable(name="codigo") String codigo) {

inventarioServiceImpl.deleteInventarioById(codigo);

}

}

# Clase DemoAppApplication.java package com.example.demo\_app

package com.example.demo\_app;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.Bean;

import org.springframework.web.servlet.config.annotation.ViewControllerRegistry;

import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;

@SpringBootApplication

public class DemoAppApplication {

public static void main(String[] args) {

SpringApplication.run(DemoAppApplication.class, args);

}

@Bean

public WebMvcConfigurer forwardToIndex() {

return new WebMvcConfigurer() {

@Override

public void addViewControllers(ViewControllerRegistry registry){

registry.addViewController("/").setViewName(

"forward:/index.html");

}

};

}

}

# Clase ServlerInitializer.java package com.example.demo\_app

package com.example.demo\_app;

import org.springframework.boot.builder.SpringApplicationBuilder;

import org.springframework.boot.web.servlet.support.SpringBootServletInitializer;

public class ServletInitializer extends SpringBootServletInitializer {

@Override

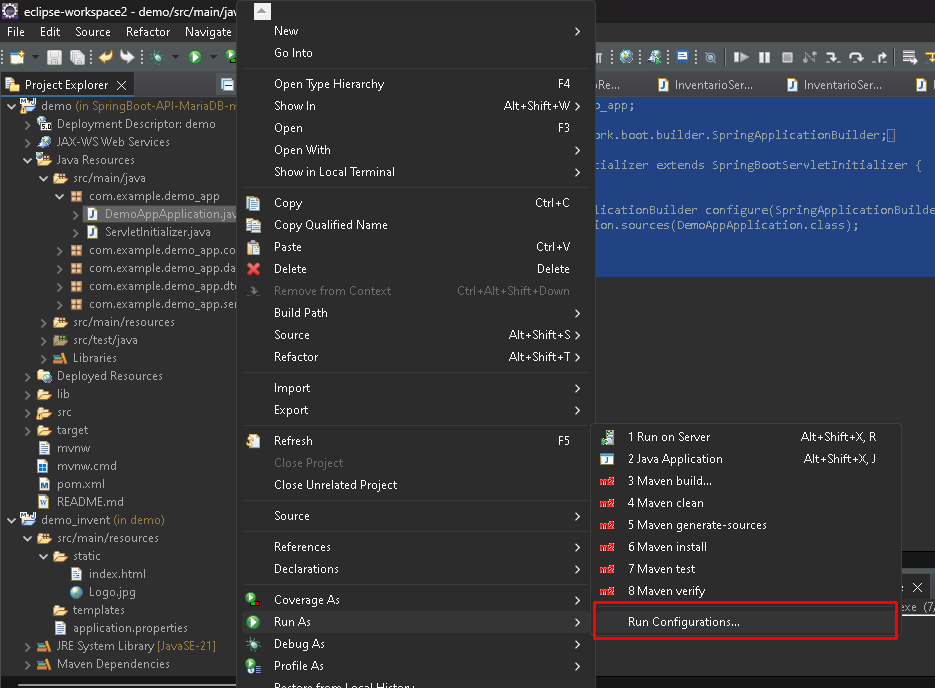
protected SpringApplicationBuilder configure(SpringApplicationBuilder application) {

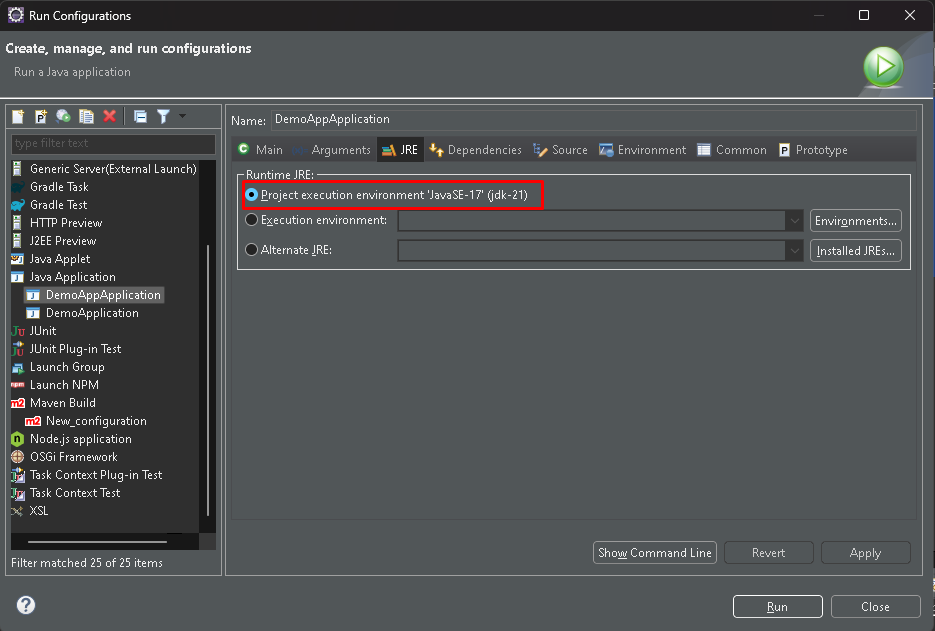
return application.sources(DemoAppApplication.class);

}

}

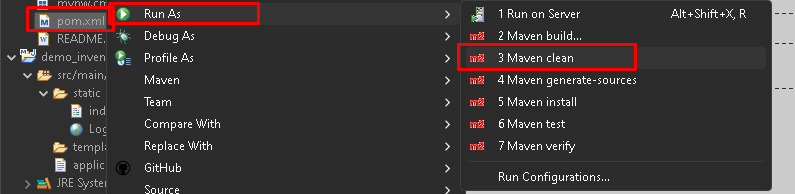
# 12.Se valida que el Maven tenga configurado la versión de Java correcta para ejecutarse.

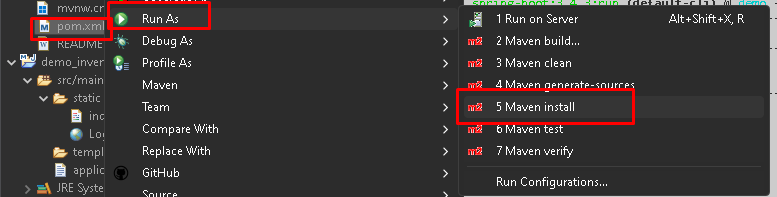




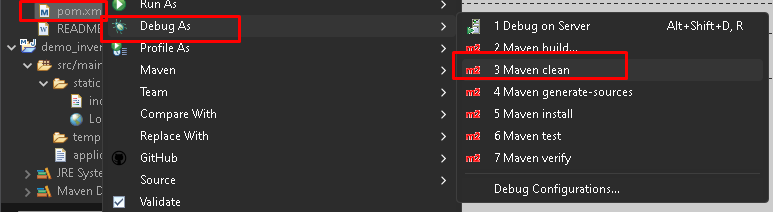
# 13.Cada vez que se haga cambio en algún archivo, presionar Ctrl+Shift+O y Ctrl+S para guardar cambios.

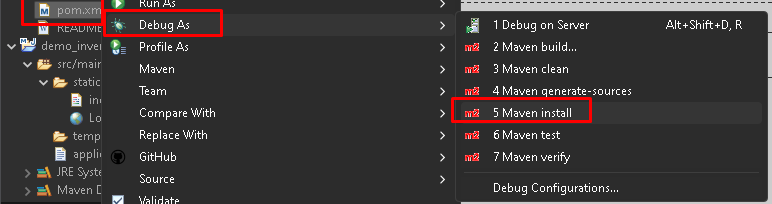
# 14.Realizar sobre el pom la compilación del proyecto entero usando Maven Clean y luego Maven Install.



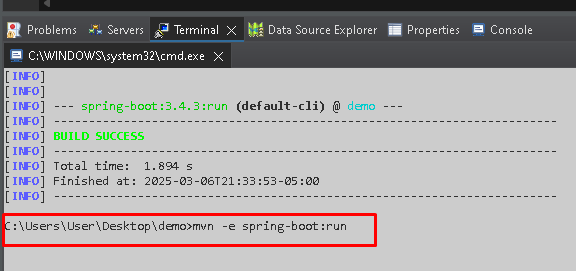


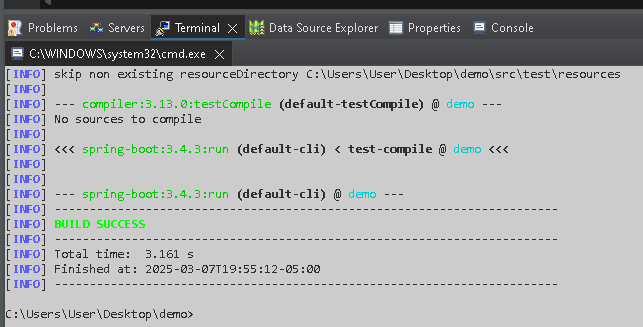
También se puede realizar debug utilizando la opción “Debug As” de Maven



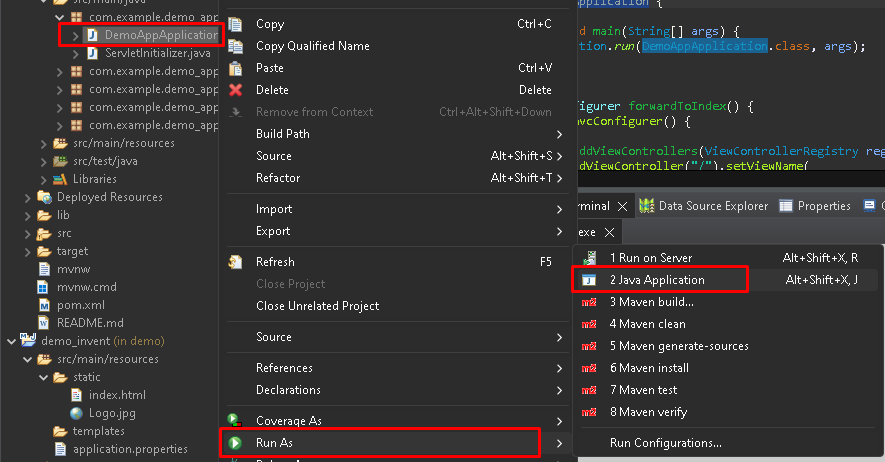


# 15.Se puede probar la correcta ejecución del proyecto de Spring Boot utilizando el comando mvn -e spring-boot:run

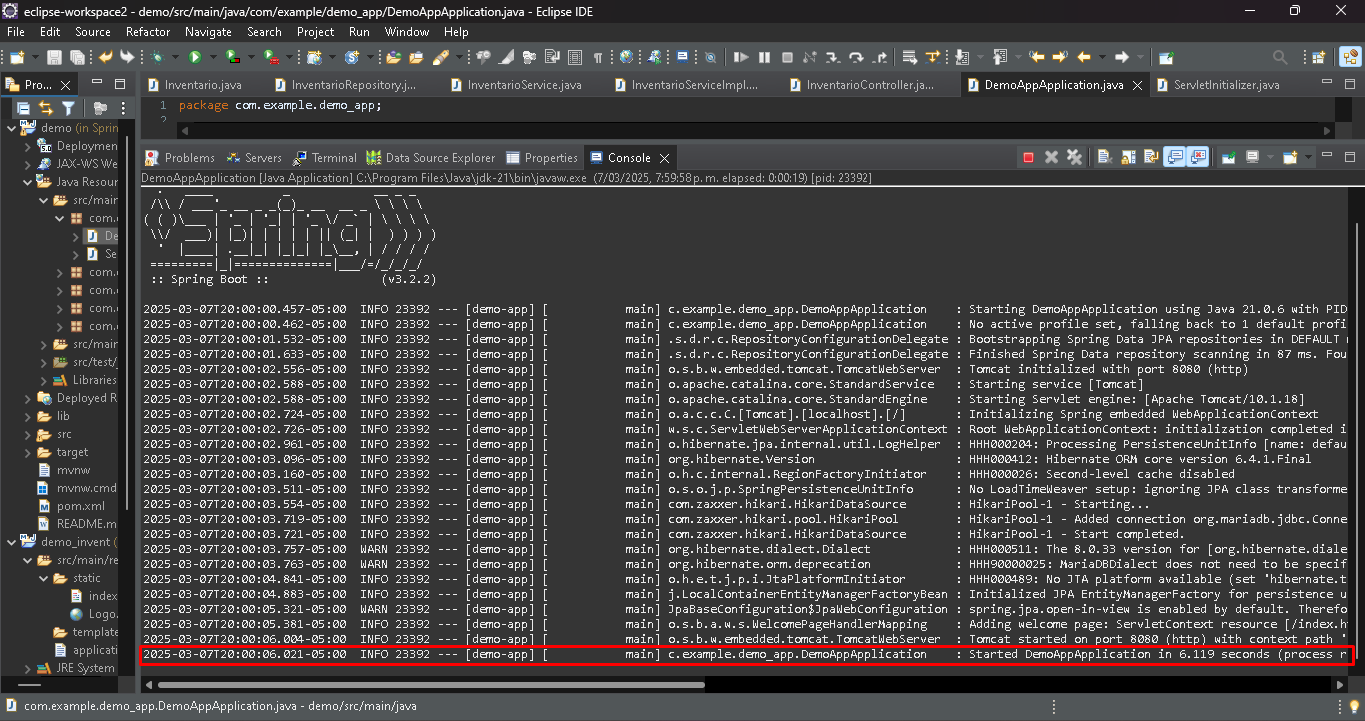




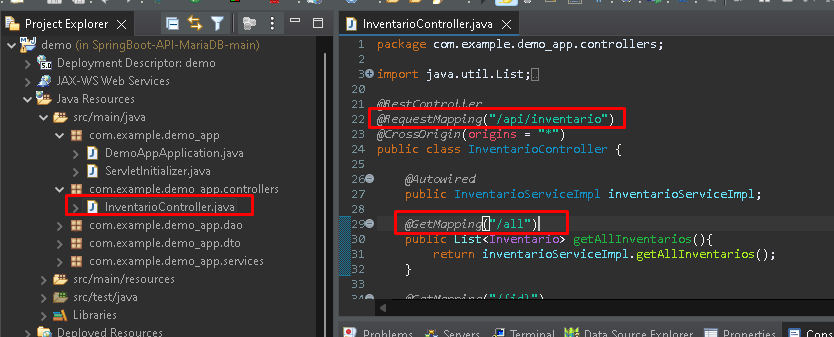
# 16.Para ejecutar el proyecto se hace clic derecho sobre la clase “DemoAppApplication.java” y se ejecuta “Run As – Java Application”.

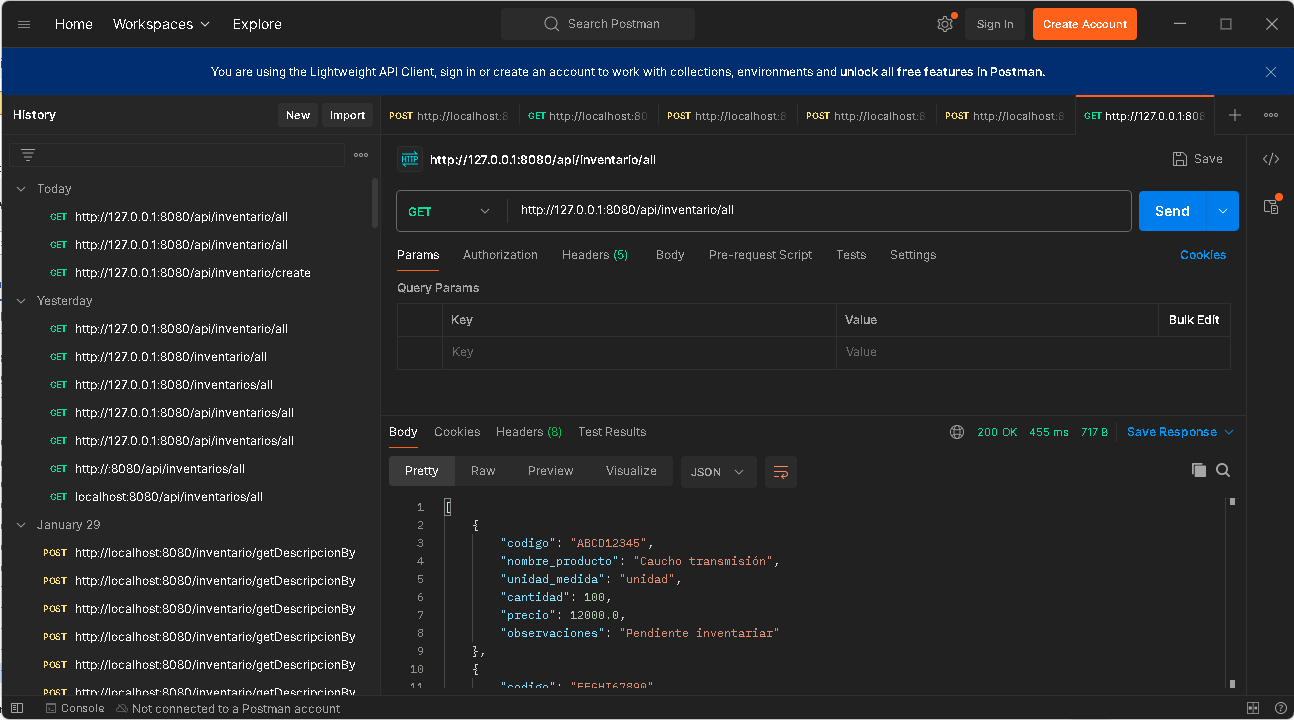


Validar que en la consola se inicie el servidor tomcat embebido



# 17. Se puede probar si está habilitado utilizando el postman con la dirección que esté configurada en la clase controller.





Ejemplo del response:

[

    {

        "codigo": "ABCD12345",

        "nombre\_producto": "Caucho transmisión",

        "unidad\_medida": "unidad",

        "cantidad": 100,

        "precio": 12000.0,

        "observaciones": "Pendiente inventariar"

    },

    {

        "codigo": "EFGHI67890",

        "nombre\_producto": "Bomba hidráulica",

        "unidad\_medida": "unidad",

        "cantidad": 10,

        "precio": 150000.0,

        "observaciones": "Pendiente inventariar"

    },

    {

        "codigo": "EFGSH002345",

        "nombre\_producto": "Guaya Hidraulica inductor",

        "unidad\_medida": "unidad",

        "cantidad": 10,

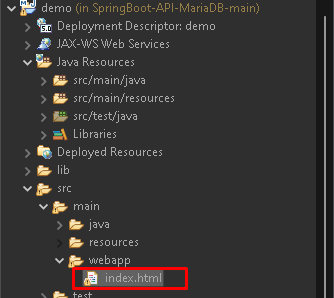
        "precio": 0.0,

        "observaciones": null

    }

]

# 18. Se puede probar con un archivo HTML para el frontend del proyecto Spring Boot en la carpeta webapp



Ejemplo de código index.html

<!**DOCTYPE** html>

<**html** lang=*"es"*>

<**head**>

<**meta** charset=*"UTF-8"*>

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

<**title**>Gestión de Inventario</**title**>

<**style**>

body {

font-family: *Arial, sans-serif*;

max-width: *800px*;

margin: *0 auto*;

padding: *20px*;

}

form {

display: *flex*;

flex-direction: *column*;

gap: *10px*;

}

input, button {

padding: *5px*;

}

table {

width: *100%*;

border-collapse: *collapse*;

margin-top: *20px*;

}

th, td {

border: *1px solid #ddd*;

padding: *8px*;

text-align: *left*;

}

th {

background-color: *#f2f2f2*;

}

header {

width:*100%*; /\* Establecemos que el header abarque el 100% del documento \*/

overflow:*hidden*; /\* Eliminamos errores de float \*/

background:*#252932*;

margin-bottom:*20px*;

}

.wrapper {

width:*90%*; /\* Establecemos que el ancho sera del 90% \*/

max-width:*1000px*; /\* Aqui le decimos que el ancho máximo sera de 1000px \*/

margin:*auto*; /\* Centramos los elementos \*/

overflow:*hidden*; /\* Eliminamos errores de float \*/

}

header .logo {

color:*#f2f2f2*;

font-size:*50px*;

line-height:*50px*;

float:*left*;

}

header nav {

float:*right*;

line-height:*50px*;

}

header nav a {

display:*inline-block*;

color:*#fff*;

text-decoration:*none*;

padding:*10px 20px*;

line-height:*normal*;

font-size:*20px*;

font-weight:*bold*;

-webkit-transition:*all 500ms ease*;

-o-transition:*all 500ms ease*;

transition:*all 500ms ease*;

}

header nav a:hover {

background:*#f56f3a*;

border-radius:*50px*;

}

</**style**>

<**link** href=*"https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"* rel=*"stylesheet"*>

</**head**>

<**body**>

<**header**>

<**div** class=*"wrapper"*>

<**div** class=*"logo"*>Distribuidora

<!--<a>

<img src="Logo.jpg" alt="logo" />

</a>

</div>-->

<**nav**>

<**a** href=*"#"*>Inventario</**a**>

<**a** href=*"#"*>Facturación</**a**>

<**a** href=*"#"*>Reportes</**a**>

<!--<a href="#"></a>-->

</**nav**>

</**div**>

</**header**>

<**h1**>Gestión de Inventario</**h1**>

<**form** id=*"inventarioForm"*>

<!-- <input type="hidden" id="codigo">-->

<**input** type=*"text"* id=*"codigo"* placeholder=*"Código Producto"* required class=*"form-control"* aria-label=*"Small"* aria-describedby=*"inputGroup-sizing-sm"*>

<**input** type=*"text"* id=*"nombre\_producto"* placeholder=*"Nombre Producto"* required>

<**input** type=*"text"* id=*"unidad\_medida"* placeholder=*"Unidad de Medida"* required>

<**input** type=*"number"* id=*"cantidad"* placeholder=*"Cantidad"* required>

<**input** type=*"number"* id=*"precio"* placeholder=*"Precio"* required>

<**input** type=*"text"* id=*"observaciones"* placeholder=*"Observaciones"* required>

<**button** type=*"submit"* class=*"btn btn-info"*>Guardar</**button**>

</**form**>

<**table** id=*"inventariosTable"*>

<**thead**>

<**tr**>

<**th**>Codigo</**th**>

<**th**>Nombre\_Producto</**th**>

<**th**>Unidad\_Medida</**th**>

<**th**>Cantidad</**th**>

<**th**>Precio</**th**>

<**th**>Observaciones</**th**>

</**tr**>

</**thead**>

<**tbody**></**tbody**>

</**table**>

<**script**>

**const** API\_URL = 'http://localhost:8080/api/inventario';

**const** form = document.getElementById('inventarioForm');

**const** table = document.getElementById('inventariosTable').getElementsByTagName('tbody')[0];

// Cargar usuarios al iniciar la página

fetchInventarios();

form.addEventListener('submit', (e) => {

e.preventDefault();

**const** inventario = {

codigo: document.getElementById('codigo').value,

nombre\_producto: document.getElementById('nombre\_producto').value,

unidad\_medida: document.getElementById('unidad\_medida').value,

cantidad: document.getElementById('cantidad').value,

precio: document.getElementById('precio').value,

observaciones: parseFloat(document.getElementById('observaciones').value)

};

**if** (inventario.codigo) {

//updateInventario(inventario);

createInventario(inventario);

} /\*else {

createInventario(inventario);

}\*/

});

**function** fetchInventarios() {

fetch(`${API\_URL}/all`)

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

.then(inventarios => {

table.innerHTML = '';

inventarios.forEach(inventario => {

**const** row = table.insertRow();

row.innerHTML = `

<td>${inventario.codigo}</td>

<td>${inventario.nombre\_producto}</td>

<td>${inventario.unidad\_medida}</td>

<td>${inventario.cantidad}</td>

<td>${inventario.precio}</td>

<td>${inventario.observaciones}</td>

<td>

<button onclick="editInventario(${inventario.codigo})">Editar</button>

<button onclick="deleteInventario(${inventario.codigo})">Eliminar</button>

</td>

`;

});

});

}

**function** createInventario(inventario) {

fetch(`${API\_URL}/create`, {

method: 'POST',

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

body: JSON.stringify(inventario)

})

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

.then(() => {

form.reset();

fetchInventarios();

});

}

**function** updateInventario(inventario) {

fetch(`${API\_URL}/update/${inventario.codigo}`, {

method: 'PUT',

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

body: JSON.stringify(inventario)

})

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

.then(() => {

form.reset();

document.getElementById('codigo').value = '';

fetchInventarios();

});

}

**function** deleteInventario(id) {

fetch(`${API\_URL}/**delete**/${codigo}`, { method: 'DELETE' })

.then(() => fetchInventarios());

}

**function** editInventario(id) {

fetch(`${API\_URL}/all`)

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

.then(inventarios => {

**const** inventario = inventarios.find(u => u.codigo === codigo);

document.getElementById('codigo').value = inventario.id;

document.getElementById('nombre\_producto').value = inventario.nombre\_producto;

document.getElementById('unidad\_medida').value = inventario.unidad\_medida;

document.getElementById('cantidad').value = inventario.cantidad;

document.getElementById('precio').value = inventario.precio;

document.getElementById('observaciones').value = inventario.observaciones;

});

}

</**script**>

<**script** src=*"https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"*></**script**>

</**body**>

</**html**>