
# FACULTAD DE INGENIERÍA Y ARQUITECTURA

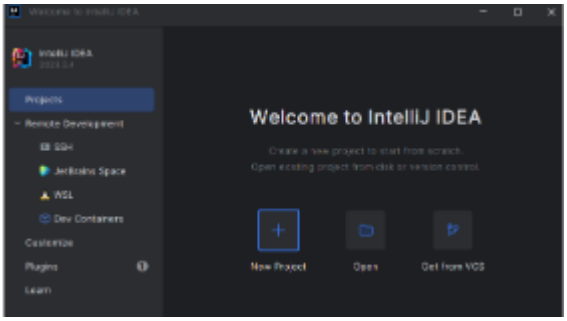
## Escuela Profesional de Ingeniería de Sistemas

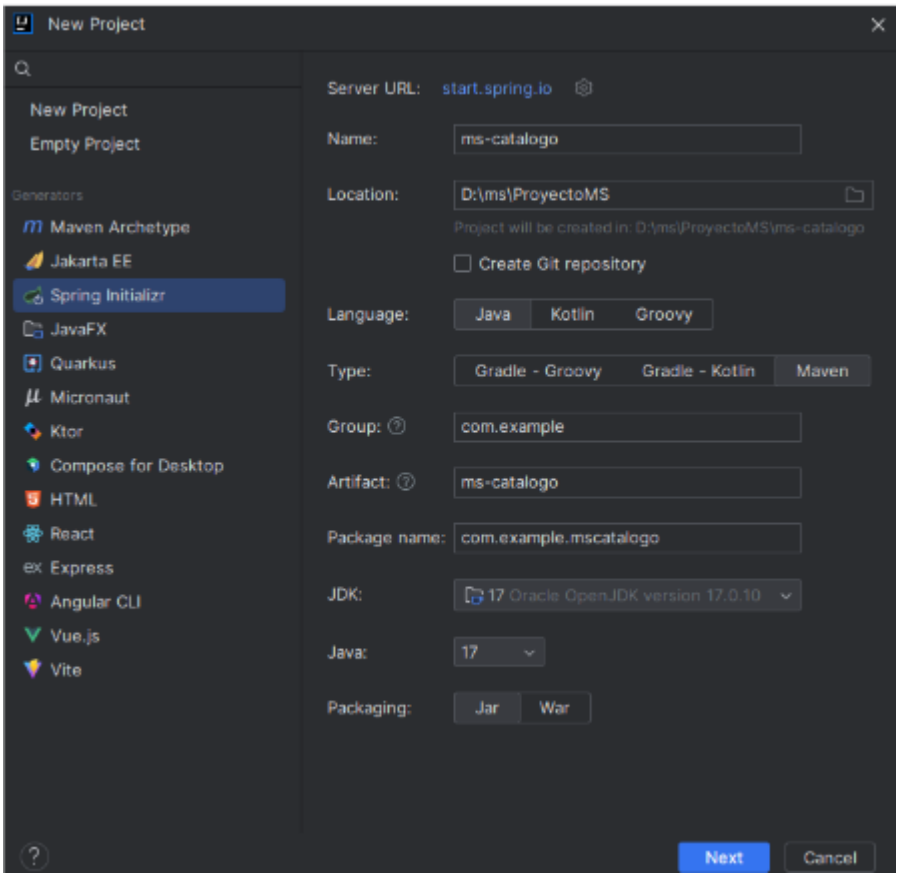
## Curso: Desarrollo de Aplicaciones Distribuidas

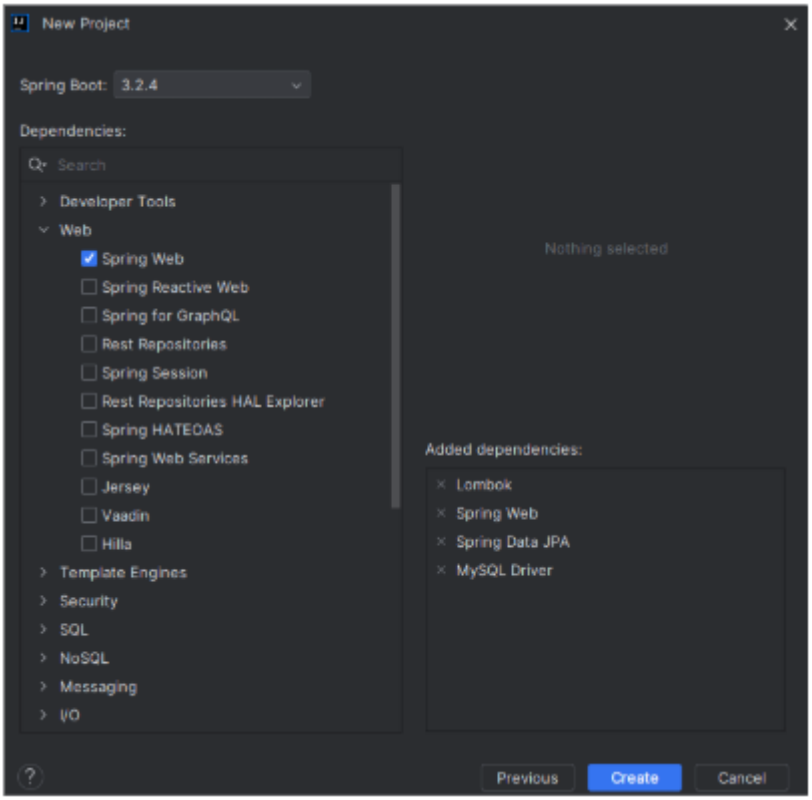
**Estudiante: William David Ortiz Mamani**

# GUÍA DE IMPLEMENTACIÓN DE MICROSERVICIOS

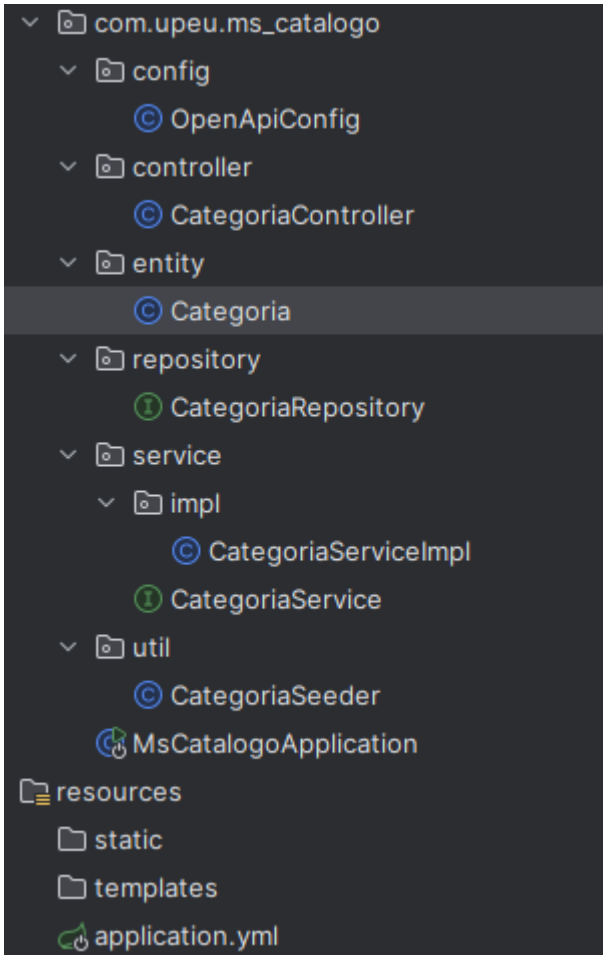
Crear proyecto


Welcome to IntelliJ IDEA
New Project
Open
Get from VCS


New Project
New Project
Empty Project
Generators
Maven Archetype
Jakarta EE
Spring Initializr
JavaFX
Quarkus
Micronaut
Ktor
Compose for Desktop
HTML
React
Express
Angular CLI
Vue.js
Vite
Server URL: start.spring.io
Name: ms-catalogo
Location: D:\ms\ProyectoMS
Project will be created in: D:\ms\ProyectoMS\ms-catalogo
Create Git repository
Language: Java Kotlin Groovy
Type: Gradle - Groovy Gradle - Kotlin Maven
Group: com.example
Artifact: ms-catalogo
Package name: com.example.mscatalogo
JDK: 17 Oracle OpenJDK version 17.0.10
Java: 17
Packaging: Jar War
Next
Cancel




ESTRUCTURA DEL ms_catalogo



Crear el entity Categoria

```
package com.upeu.ms_catalogo.entity;

import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;

@Entity
public class Categoria {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   private Integer id;
   private String nombre;

   public Integer getId() {
      return id;
   }

   public void setId(Integer id) {
      this.id = id;
   }

   public String getNombre() {
      return nombre;
   }

   public void setNombre(String nombre) {
      this.nombre = nombre;
   }

   public Categoria(Integer id, String nombre) {
      this.id = id;
      this.nombre = nombre;
   }

   public Categoria() {

   }

   @Override
   public String toString() {
      return "Categoria{" +
            "id=" + id +
            ", nombre='" + nombre + '\'' +
            '}';
   }
}
```

Repository

```
package com.upeu.ms_catalogo.repository;

import com.upeu.ms_catalogo.entity.Categoria;
import org.springframework.data.jpa.repository.JpaRepository;

public interface CategoriaRepository extends JpaRepository<Categoria, Integer> {
}
```

CategoriaService

```
package com.upeu.ms_catalogo.service;

import com.upeu.ms_catalogo.entity.Categoria;

import java.util.List;
import java.util.Optional;

public interface CategoriaService {
   List<Categoria> listar();

   Optional<Categoria> buscar(Integer id);

   Categoria guardar(Categoria categoria);

   Categoria actualizar(Integer id, Categoria categoria);

   void eliminar(Integer id);

}
```

CategoriaServiceImpl

```
package com.upeu.ms_catalogo.service.impl;

import com.upeu.ms_catalogo.entity.Categoria;
import com.upeu.ms_catalogo.repository.CategoriaRepository;
import com.upeu.ms_catalogo.service.CategoriaService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;
import java.util.Optional;

@Service
public class CategoriaServiceImpl implements CategoriaService {
   @Autowired
   private CategoriaRepository categoriaRepository;

   @Override
   public List<Categoria> listar() {
```

```
        return categoriaRepository.findAll();
    }

    @Override
    public Optional<Categoria> buscar(Integer id) {
        return categoriaRepository.findById(id);
    }

    @Override
    public Categoria guardar(Categoria categoria) {
        return categoriaRepository.save(categoria);
    }

    @Override
    public Categoria actualizar(Integer id, Categoria categoria) {
        categoria.setId(id);
        return categoriaRepository.save(categoria);
    }

    @Override
    public void eliminar(Integer id) {
        categoriaRepository.deleteById(id);
    }
}
```

CategoriaController

```
package com.upeu.ms_catalogo.controller;

import com.upeu.ms_catalogo.entity.Categoria;
import com.upeu.ms_catalogo.service.CategoriaService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

import java.util.List;
import java.util.Optional;

@RestController
@RequestMapping("/categorias")
public class CategoriaController {
    @Autowired
    private CategoriaService categoriaService;

    @RequestMapping
    public List<Categoria> listar() {
        return categoriaService.listar();
    }

    @RequestMapping("/{id}")
    public Optional<Categoria> buscar(@PathVariable Integer id) {
        return categoriaService.buscar(id);
    }
```

```
    @PostMapping
    public Categoria guardar(@RequestBody Categoria categoria) {
        System.out.println(categoria);
        return categoriaService.guardar(categoria);
    }

    @PutMapping("/{id}")
    public Categoria actualizar(@PathVariable Integer id, @RequestBody Categoria categoria)
{
        return categoriaService.actualizar(id, categoria);
    }

    @DeleteMapping("/{id}")
    public void eliminar(@PathVariable Integer id) {
        categoriaService.eliminar(id);
    }
}
```

En el config / OpenApiConfig

```
package com.upeu.ms_catalogo.config;

import io.swagger.v3.oas.models.OpenAPI;
import io.swagger.v3.oas.models.info.Info;
import io.swagger.v3.oas.models.info.License;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

@Configuration
public class OpenApiConfig {
    @Bean
    public OpenAPI customOpenAPI() {
        return new OpenAPI()
                .info(new Info()
                        .title("OPEN API MICROSERVICIO CATÁLOGO")
                        .version("0.0.1")
                        .description("Servicios web catálogo")
                        .termsOfService("http://swagger.io/terms")
                        .license(new License().name("Apache 2.0").url("http://springdoc.org"))
                );
    }
}
```

En el util/ CategoriaSeeder

```
package com.upeu.ms_catalogo.util;

import com.upeu.ms_catalogo.entity.Categoria;
import com.upeu.ms_catalogo.repository.CategoriaRepository;
import org.springframework.boot.CommandLineRunner;
import org.springframework.stereotype.Component;
```

```
@Component
public class CategoriaSeeder implements CommandLineRunner {

   private final CategoriaRepository categoriaRepository;

   public CategoriaSeeder(CategoriaRepository categoriaRepository) {
      this.categoriaRepository = categoriaRepository;
   }

   @Override
   public void run(String... args) {
      // Verificamos si ya existen datos para no duplicar
      if (categoriaRepository.count() == 0) {
         Categoria cat1 = new Categoria(null, "Electrónica");
         Categoria cat2 = new Categoria(null, "Ropa");
         Categoria cat3 = new Categoria(null, "Hogar");
         Categoria cat4 = new Categoria(null, "Juguetes");
         Categoria cat5 = new Categoria(null, "Libros");
         Categoria cat6 = new Categoria(null, "Deportes");
         Categoria cat7 = new Categoria(null, "Salud");
         Categoria cat8 = new Categoria(null, "Belleza");
         Categoria cat9 = new Categoria(null, "Automotriz");
         Categoria cat10 = new Categoria(null, "Mascotas");
         Categoria cat11 = new Categoria(null, "Herramientas");
         Categoria cat12 = new Categoria(null, "Computadoras");
         Categoria cat13 = new Categoria(null, "Celulares");
         Categoria cat14 = new Categoria(null, "Música");
         Categoria cat15 = new Categoria(null, "Jardín");
         Categoria cat16 = new Categoria(null, "Alimentos");
         Categoria cat17 = new Categoria(null, "Bebidas");
         Categoria cat18 = new Categoria(null, "Electrodomésticos");
         Categoria cat19 = new Categoria(null, "Joyería");
         Categoria cat20 = new Categoria(null, "Videojuegos");

         categoriaRepository.save(cat1);
         categoriaRepository.save(cat2);
         categoriaRepository.save(cat3);
         categoriaRepository.save(cat4);
         categoriaRepository.save(cat5);
         categoriaRepository.save(cat6);
         categoriaRepository.save(cat7);
         categoriaRepository.save(cat8);
         categoriaRepository.save(cat9);
         categoriaRepository.save(cat10);
         categoriaRepository.save(cat11);
         categoriaRepository.save(cat12);
         categoriaRepository.save(cat13);
         categoriaRepository.save(cat14);
         categoriaRepository.save(cat15);
         categoriaRepository.save(cat16);
         categoriaRepository.save(cat17);
```

```
            categoriaRepository.save(cat18);
            categoriaRepository.save(cat19);
            categoriaRepository.save(cat20);

            System.out.println("Datos de categorías insertados correctamente.");
        } else {
            System.out.println("Las categorías ya existen, no se insertaron datos.");
        }
    }
}
```

En el application.yml

```
spring:
  application:
    name: ms-catalogo-service
  profiles:
    active: development
  config:
    import: optional:configserver:http://root:123456@localhost:7070
```

En el pom añadir y borrar

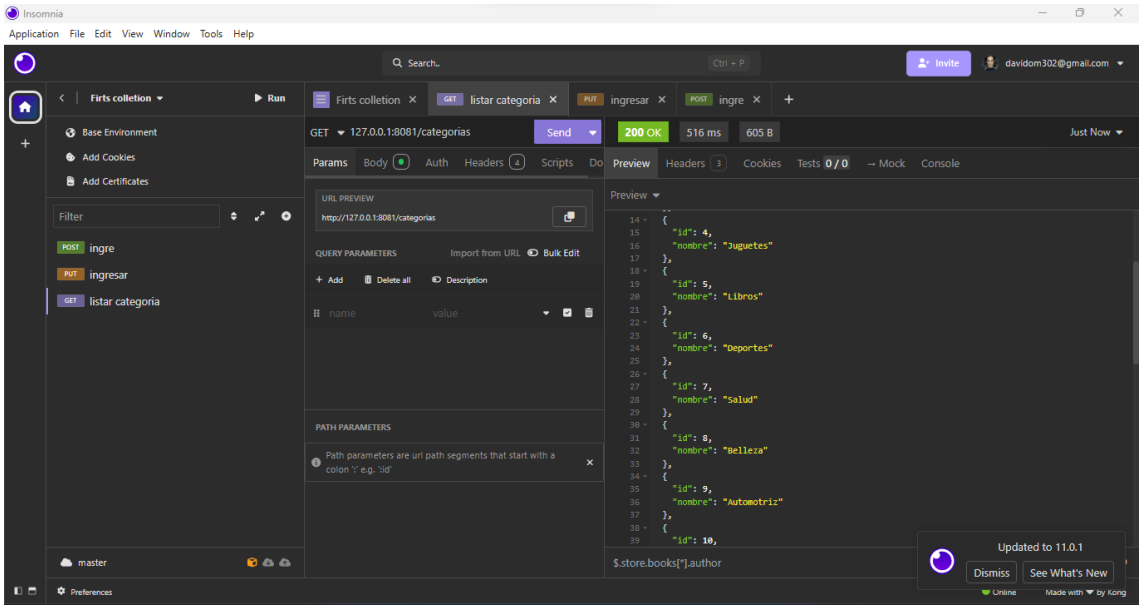
```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>3.1.3</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.upeu</groupId>
    <artifactId>ms_catalogo</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>ms_catalogo</name>
    <description>ms_catalogo</description>
    <url/>
    <licenses>
        <license/>
    </licenses>
    <developers>
        <developer/>
    </developers>
```

```
  <scm>
    <connection/>
    <developerConnection/>
    <tag/>
    <url/>
  </scm>
  <properties>
    <java.version>17</java.version>
    <spring-cloud.version>2022.0.2</spring-cloud.version>
  </properties>
  <dependencies>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
    <dependency>
      <groupId>org.springdoc</groupId>
      <artifactId>springdoc-openapi-starter-webmvc-ui</artifactId>
      <version>2.0.2</version>
    </dependency>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-validation</artifactId>
    </dependency>

    <!--<dependency>
        <groupId>com.mysql</groupId>
        <artifactId>mysql-connector-j</artifactId>
        <scope>runtime</scope>
     </dependency>-->

    <!-- Dependencia para la base de datos H2 -->
    <dependency>
      <groupId>com.h2database</groupId>
      <artifactId>h2</artifactId>
      <version>2.1.214</version>  <!-- Usa la versión más reciente -->
      <scope>runtime</scope>
    </dependency>
    <dependency>
      <groupId>org.projectlombok</groupId>
      <artifactId>lombok</artifactId>
      <optional>true</optional>
    </dependency>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-test</artifactId>
      <scope>test</scope>
    </dependency>
```

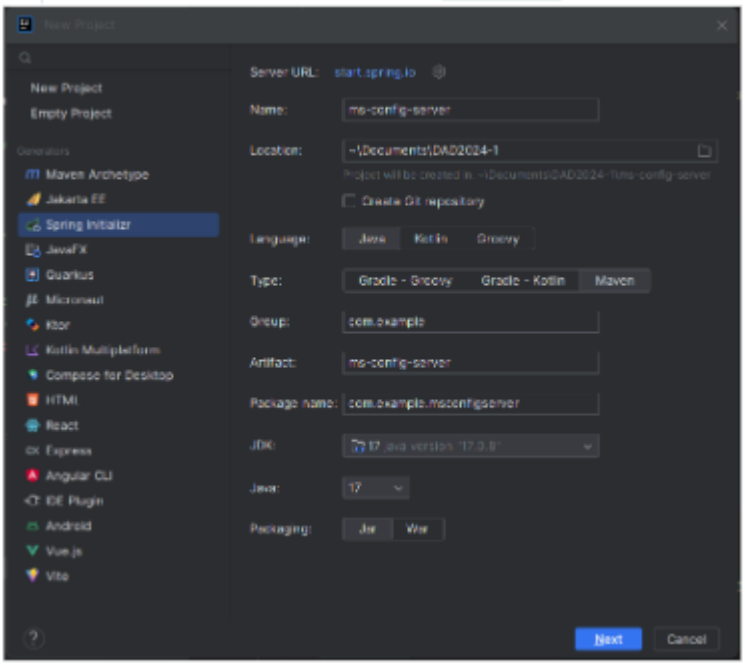
```
    <dependency>
      <groupId>org.springframework.cloud</groupId>
      <artifactId>spring-cloud-starter-config</artifactId>
      <version>4.0.4</version>
    </dependency>

  </dependencies>

  <build>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-compiler-plugin</artifactId>
        <configuration>
          <annotationProcessorPaths>
            <path>
              <groupId>org.projectlombok</groupId>
              <artifactId>lombok</artifactId>
            </path>
          </annotationProcessorPaths>
        </configuration>
      </plugin>
      <plugin>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-maven-plugin</artifactId>
        <configuration>
          <excludes>
            <exclude>
              <groupId>org.projectlombok</groupId>
              <artifactId>lombok</artifactId>
            </exclude>
          </excludes>
        </configuration>
      </plugin>
    </plugins>
  </build>

</project>
```

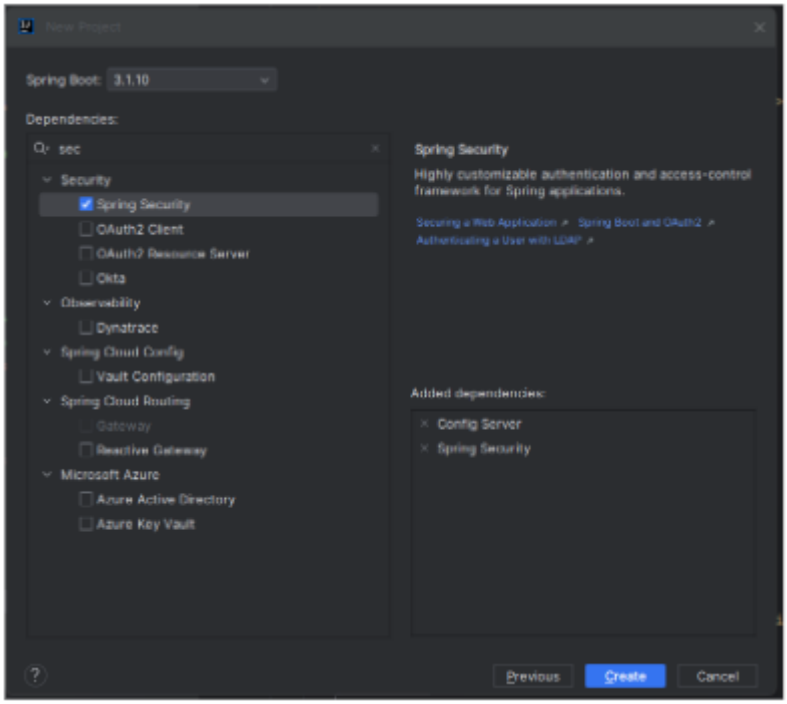
PROBAR EN EL LOCALHOST EN INSOMNIA

127.0.0.1:8081/categorias




Crear un nuevo Proyecto con el nombre ms-config-serve






En el MsConfigServeAplication añadir @EnableConfigServer

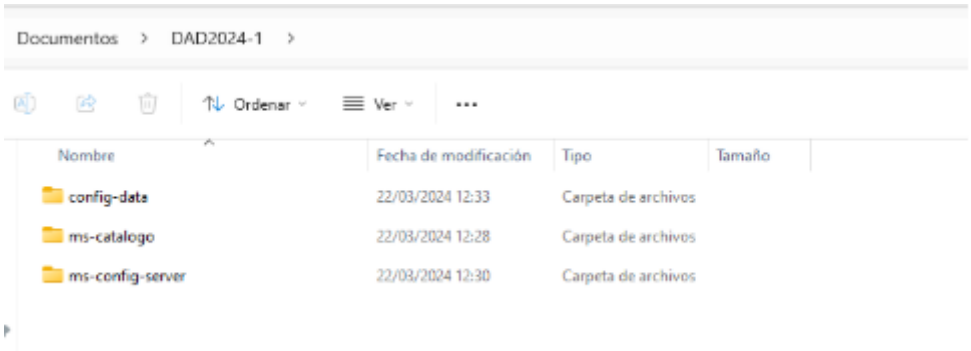
```
@SpringBootApplication
@EnableConfigServer
public class MsConfigServerApplication {

   public static void main(String[] args) {
      SpringApplication.run(MsConfigServerApplication.class, args);
   }

}
```

En el application.yml

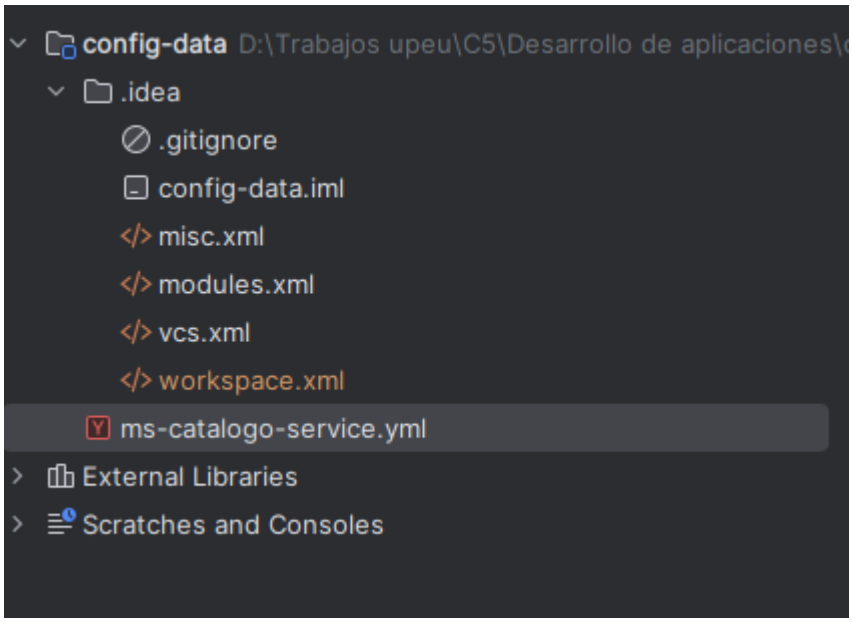
```
server:
  port: 7070

spring:
  application:
    name: config-server
  cloud:
    config:
      server:
        git:
          uri: https://github.com/302xdwill/desarrollo-de-aplicaciones.git # ruta del repositorio
git
          searchPaths: config-data
          default-label: main
  security:
    user:
```

```
    name: root
    password: 123456
```

Crear un nuevo Proyecto con el nombre config data




Crear un archivo ms-catalogo-service.yml




```
server:
  port: 8081
spring:
  datasource:
    url: jdbc:h2:mem:ms_catalogo
    driverClassName: org.h2.Driver
    username: sa
    password:
  h2:
    console:
      enabled: true
  jpa:
    database-platform: org.hibernate.dialect.H2Dialect
    hibernate:
```

```
    ddl-auto: update

springdoc:
  api-docs:
    enabled: true
  swagger-ui:
    enabled: true
    path: /doc/swagger-ui.html
```

PROBAR EN EL LOCALHOST

Localhost:8081/doc/swagger-ui.html

RESULTADO FINAL