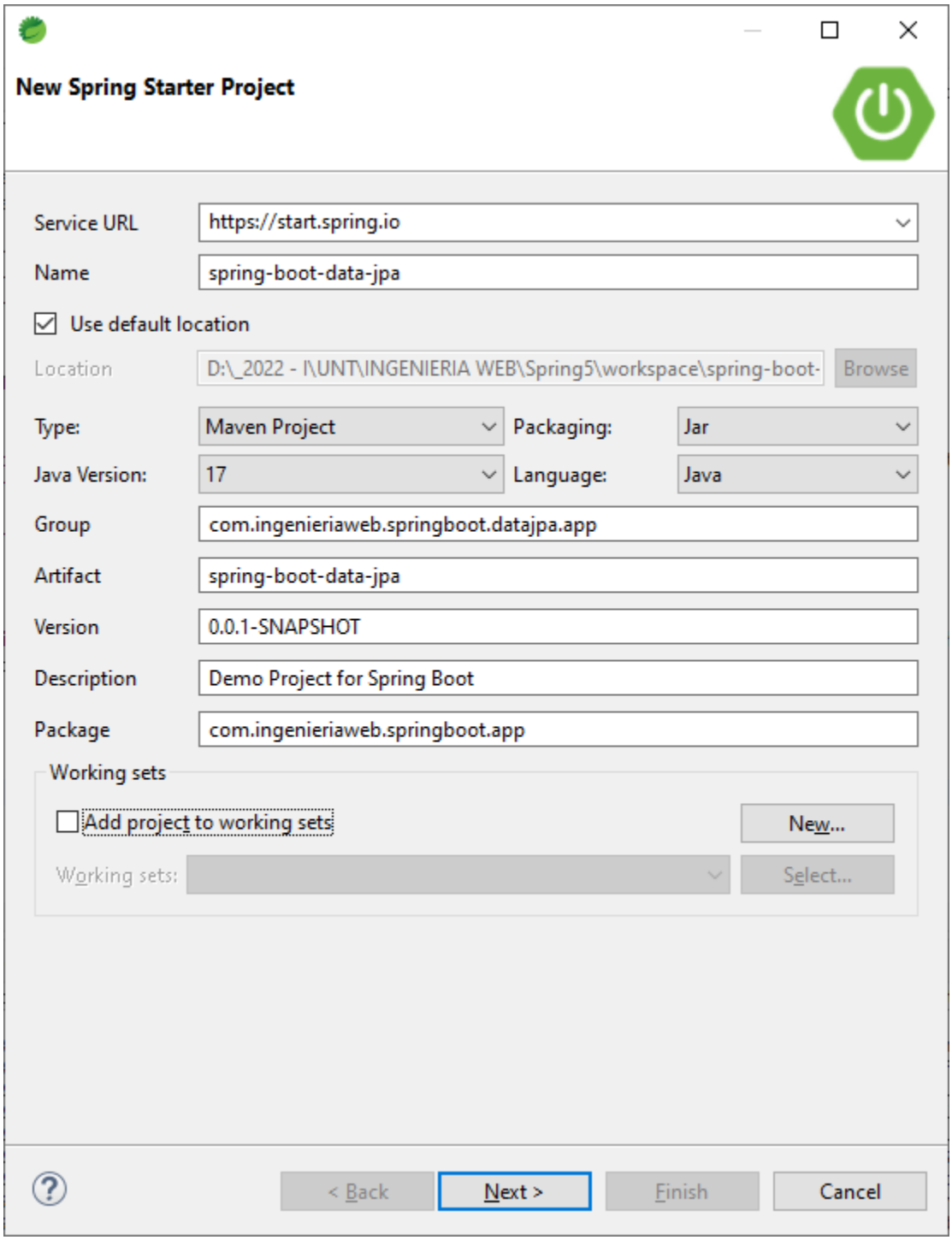
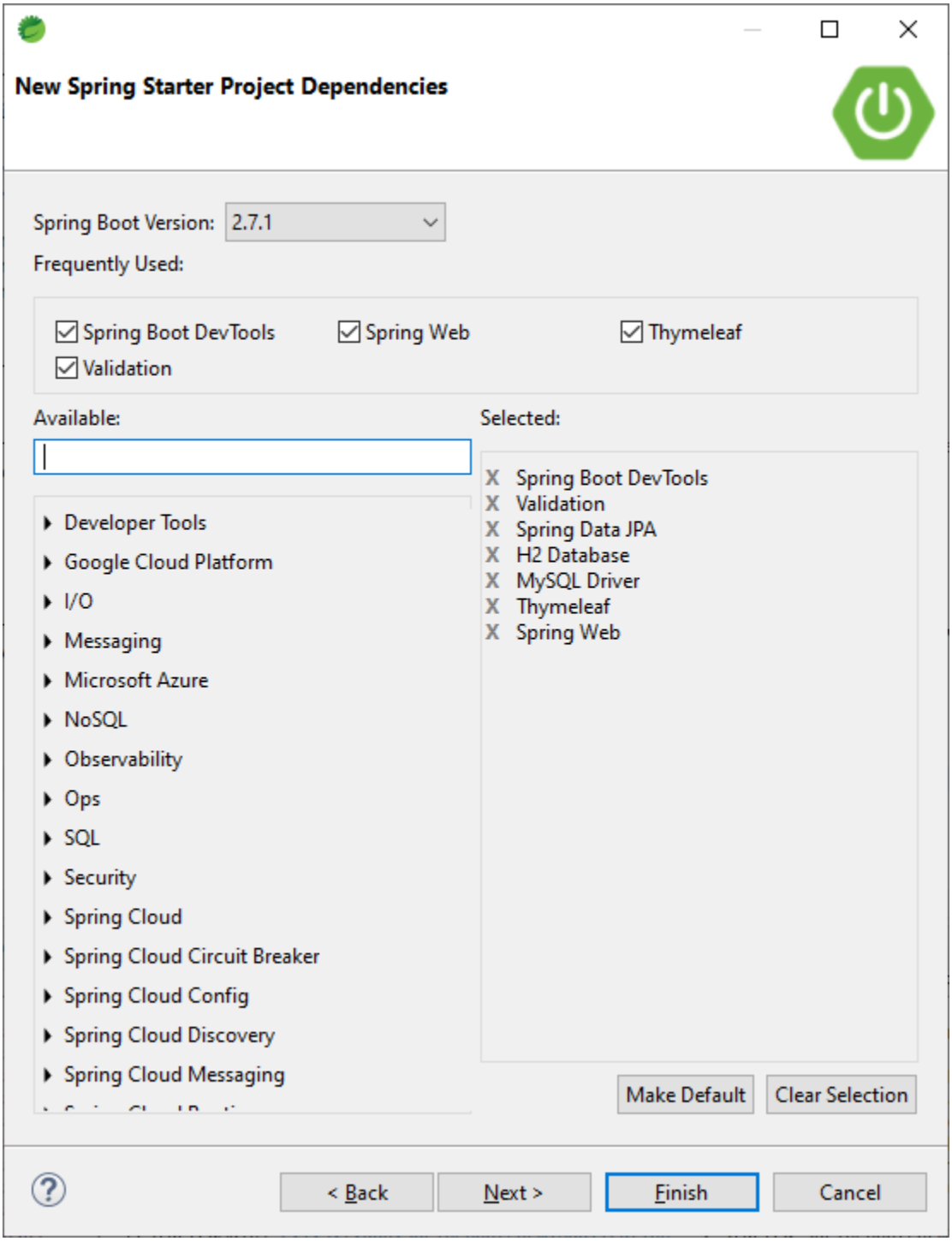
Creamos un proyecto llamado spring-boot-data-jpa

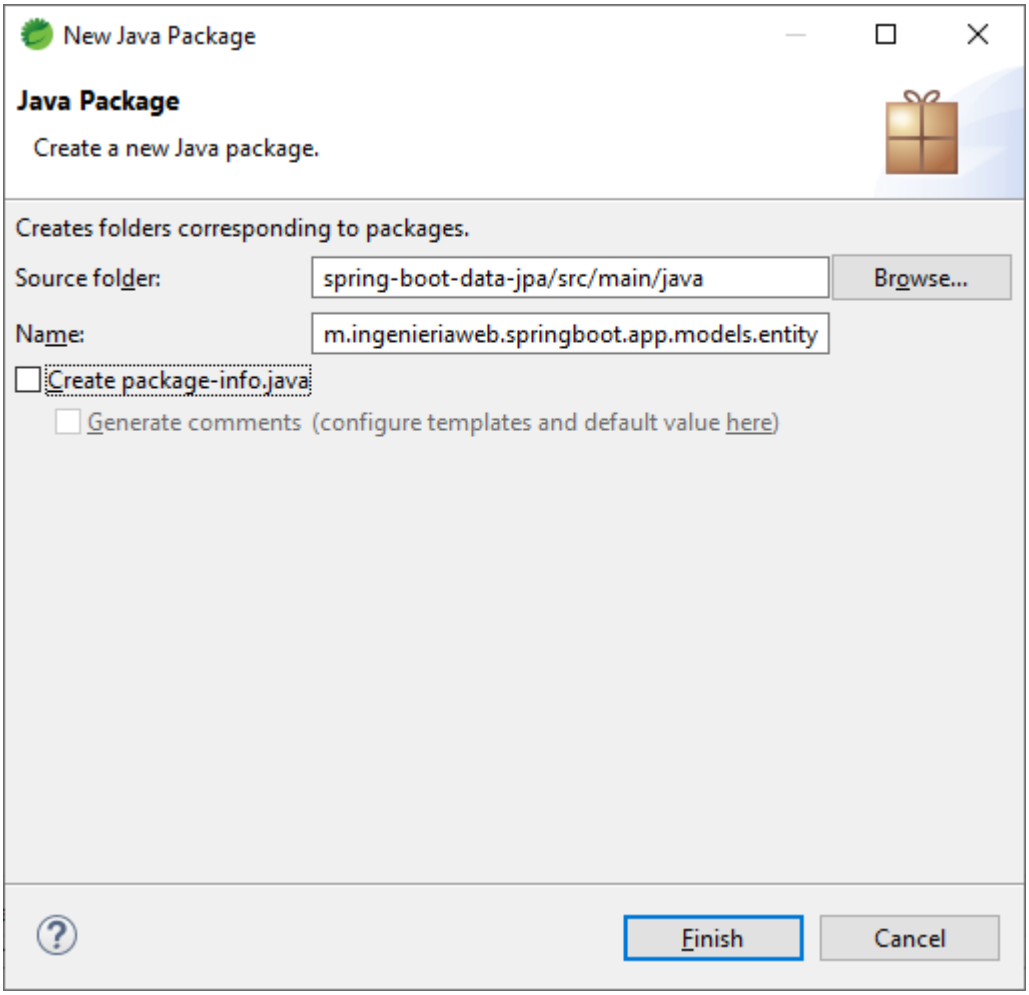


Marcar las dependencias:

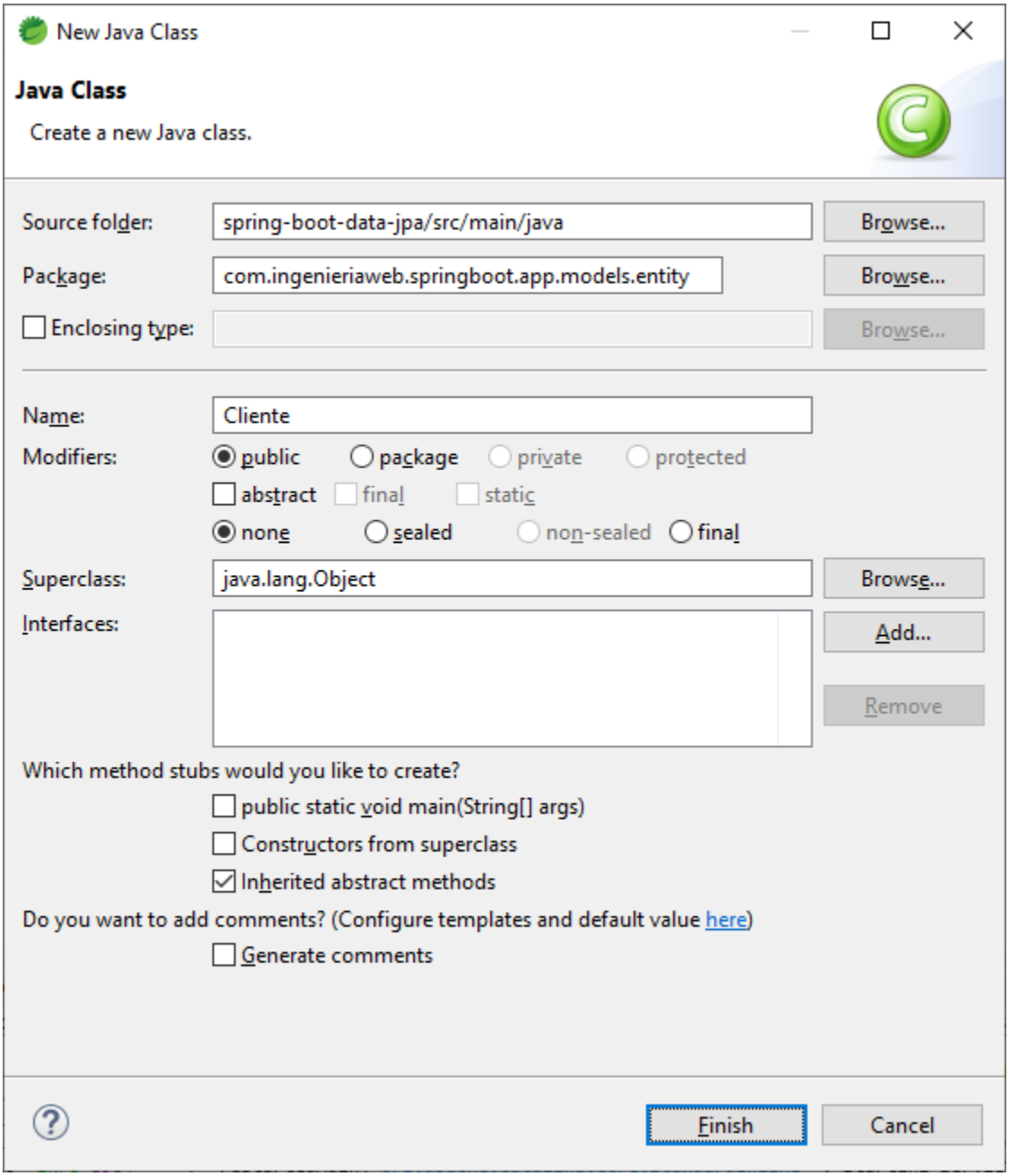


Creando la Entidad JPA anotada con @Entity

Creamos el paquete models.entity



Dentro de este paquete creamos una clase Cliente



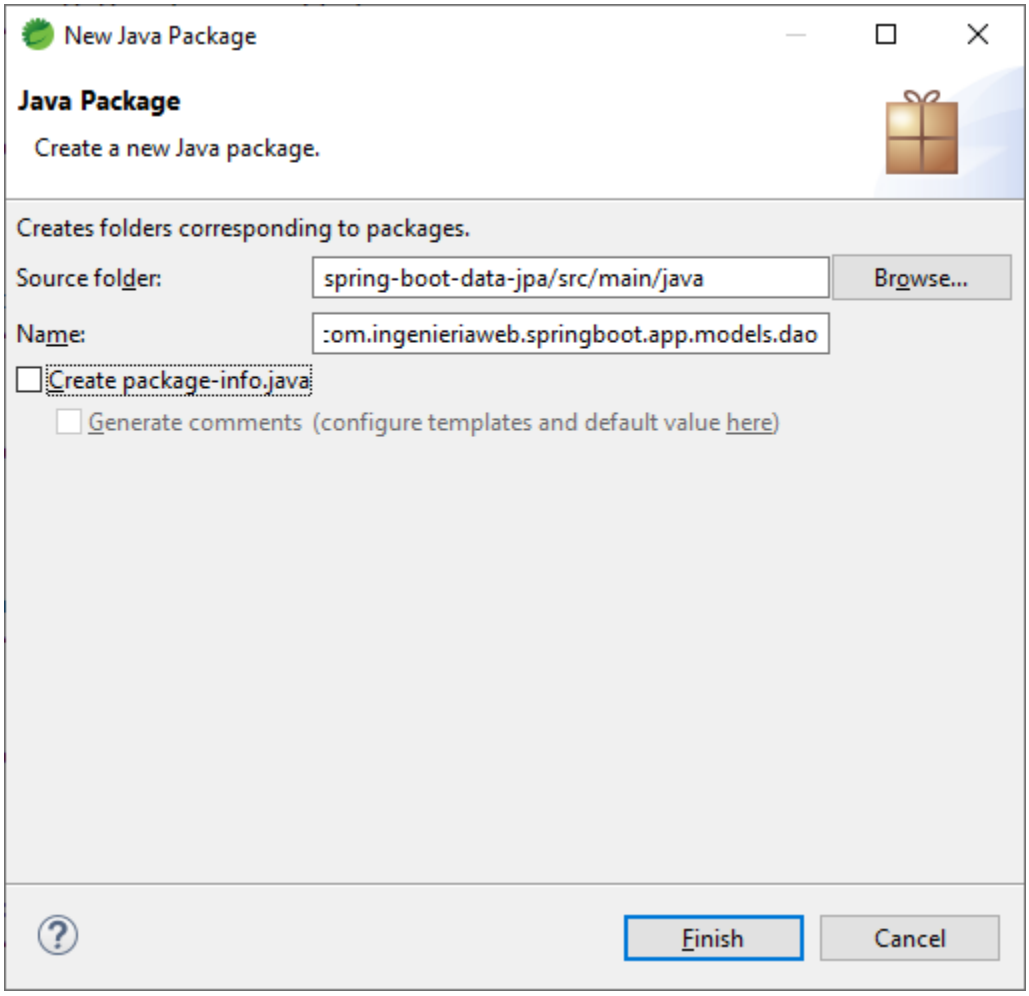
Impleentamos de Serializable y le agregamos un seriealVersionUID por defecto.

```
1  package com.ingenieriaweb.springboot.app.models.entity;
2
3  import java.io.Serializable;
4
5  import javax.persistence.Entity;
6  import javax.persistence.Table;
7
8  @Entity
9  @Table(name="clientes")
10 public class Cliente implements Serializable{
11
12     private static final long serialVersionUID = 1L;
13
14
15 }
```

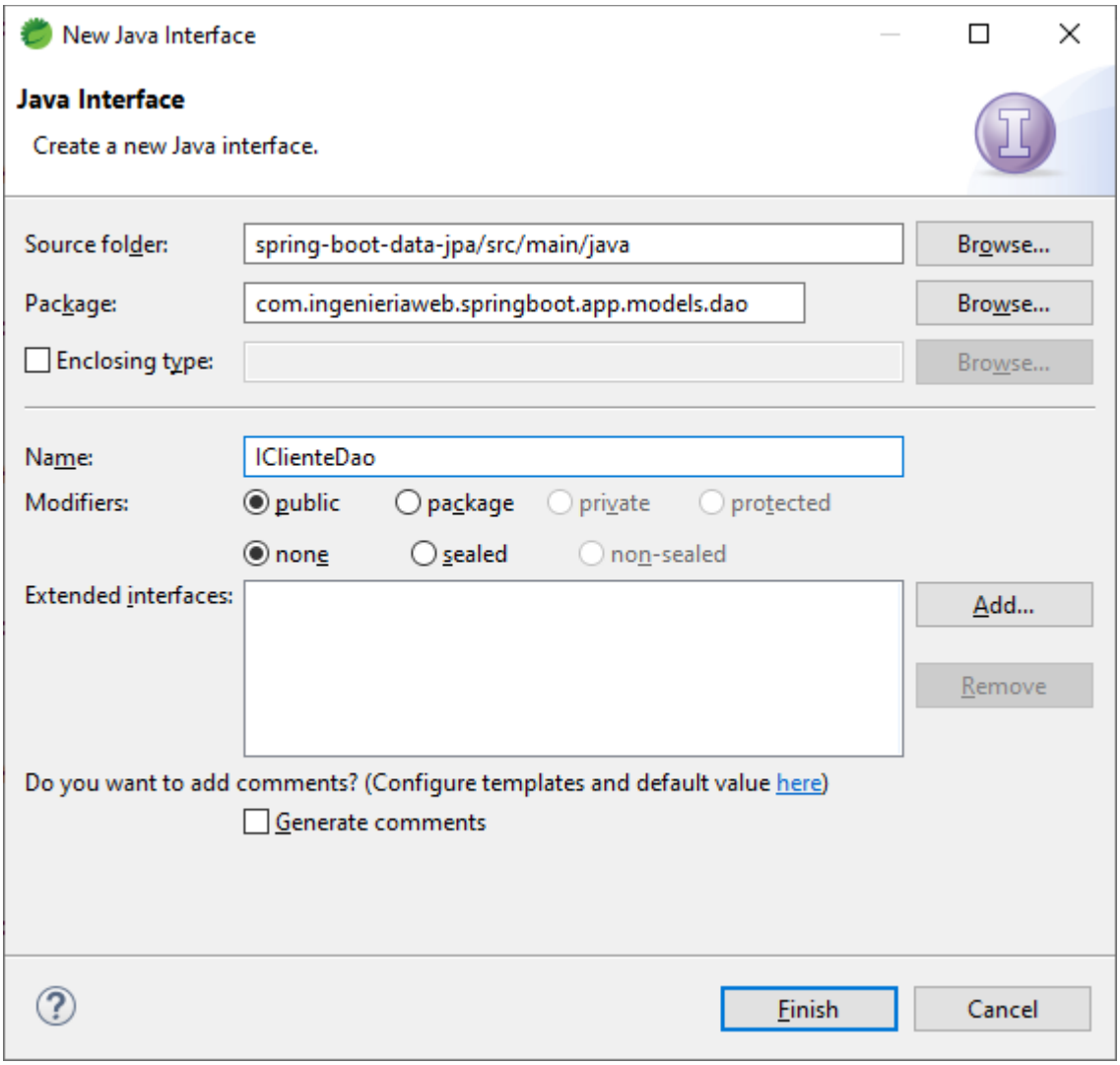
Luego colocamos sus atributos y luego colocamos sus setters y getters.

```
1  package com.ingenieriaweb.springboot.app.models.entity;
2
3  import java.io.Serializable;
4  import java.util.Date;
5  import javax.persistence.Column;
6  import javax.persistence.Entity;
7  import javax.persistence.GeneratedValue;
8  import javax.persistence.GenerationType;
9  import javax.persistence.Id;
10 import javax.persistence.Table;
11 import javax.persistence.Temporal;
12 import javax.persistence.TemporalType;
13
14 @Entity
15 @Table(name="clientes")
16 public class Cliente implements Serializable{
17     @Id
18     @GeneratedValue(strategy=GenerationType.IDENTITY)
19     private Long id;
20
21     private String nombre;
22
23     private String apellido;
24
25     private String email;
26
27     @Column(name="create_at")
28     @Temporal(TemporalType.DATE)
29     private Date createAt;
30
31     private static final long serialVersionUID = 1L;
32 }
33
```

Ahora creamos el paquete dao



Creamos la interfaz: IClienteDao



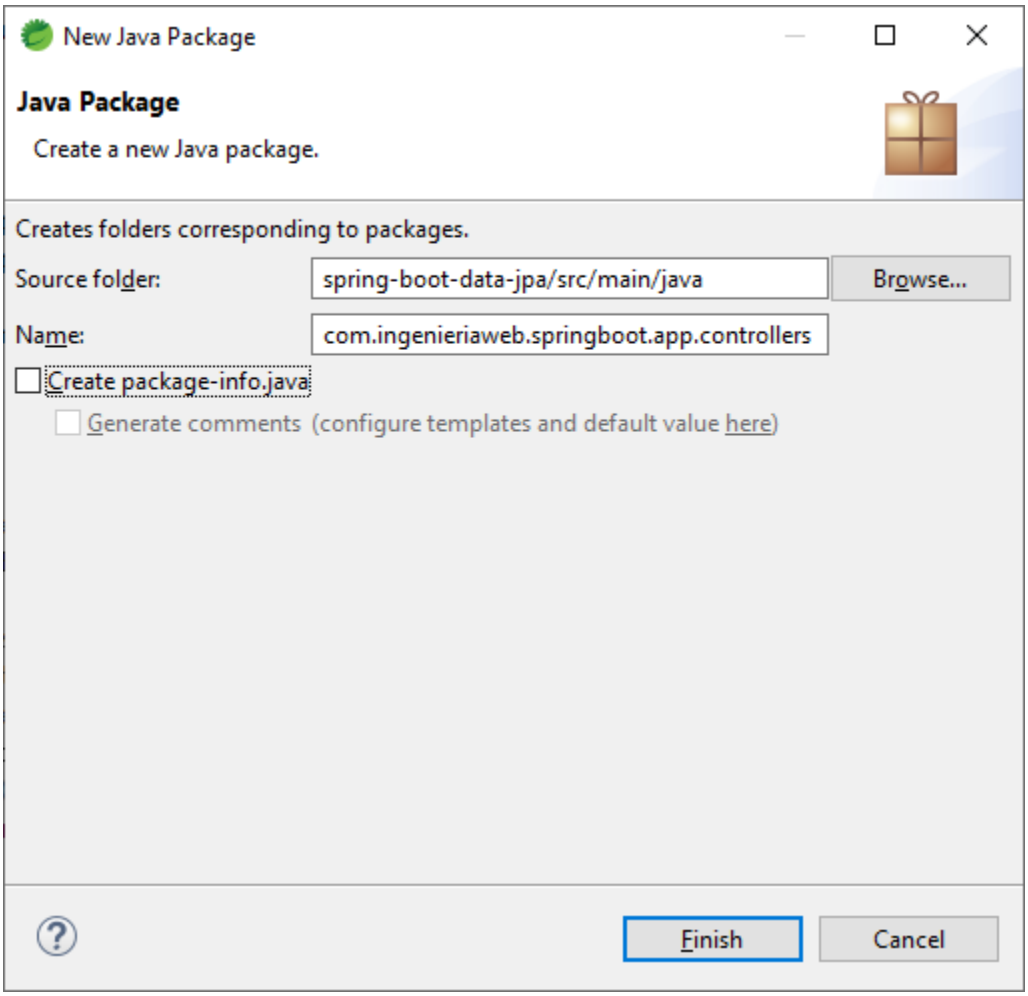
Luego creamos la clase ClienteDaoImpl que implementa de la interfaz IClienteDao

```java
package com.ingenieriaweb.springboot.app.models.dao;

import java.util.List;

import javax.persistence.EntityManager;

import org.springframework.stereotype.Repository;

import com.ingenieriaweb.springboot.app.models.entity.Cliente;

@Repository
public class ClienteDaoImpl implements IClienteDao {

    private EntityManager em;
    @Override
    public List<Cliente> findAll() {
        // TODO Auto-generated method stub
        return null;
    }

}
```

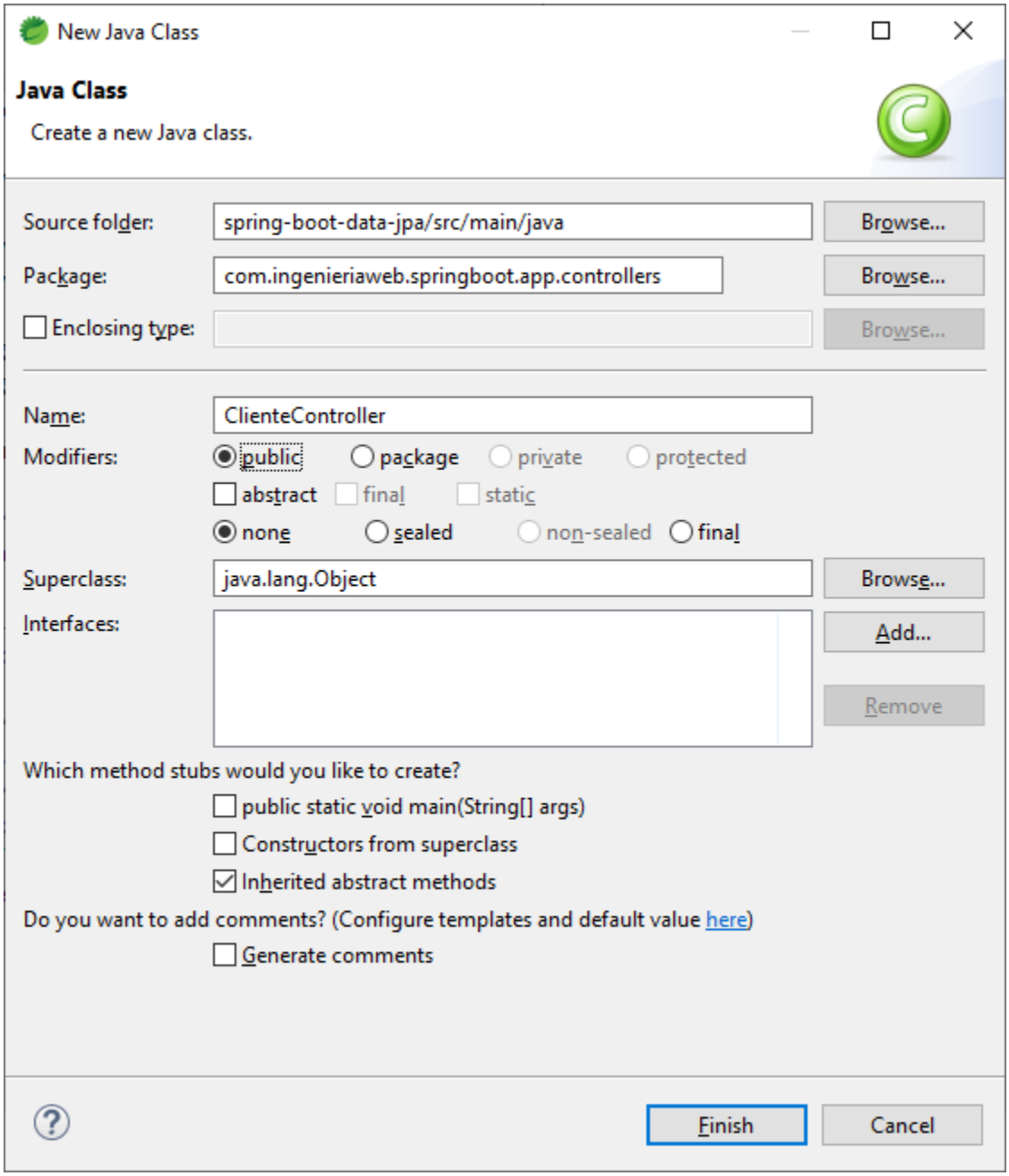
```java
package com.ingenieriaweb.springboot.app.models.dao;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.springframework.stereotype.Repository;
import org.springframework.transaction.annotation.Transactional;

import com.ingenieriaweb.springboot.app.models.entity.Cliente;

@Repository
public class ClienteDaoImpl implements IClienteDao {

    @PersistenceContext
    private EntityManager em;

    @SuppressWarnings("unchecked")
    @Transactional(readOnly=true)
    @Override
    public List<Cliente> findAll() {
        // TODO Auto-generated method stub
        return em.createQuery("from Cliente").getResultList();
    }

}
```

**Creando Controlador con la acción Handler Listar**

Creamos el paquete controllers



Creamos la clase Controladora Cliente Controller



```java
package com.ingenieriaweb.springboot.app.controllers;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

import com.ingenieriaweb.springboot.app.models.dao.IClienteDao;

@Controller
public class ClienteController {

    @Autowired
    private IClienteDao clienteDao;

    @RequestMapping(value="listar",method=RequestMethod.GET)
    public String listar(Model model) {
        model.addAttribute("titulo","listado de clientes");
        model.addAttribute("clientes",clienteDao.findAll());
        return "listar";
    }

}
```

Creando la Vista Listar.html  en resources/templates

```html
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="ISO-8859-1" />
<title th:text="${titulo}"></title>
</head>
<body>
    <h1 th:text="${titulo}"></h1>
    <table>
        <thead>
            <tr>
                <th>id</th>
                <th>nombre</th>
                <th>apellido</th>
                <th>email</th>
                <th>fecha</th>
            </tr>
        </thead>
        <tbody>
            <tr th:each="cliente: ${clientes}">
                <td th:text="${cliente.id}"></td>
                <td th:text="${cliente.nombre}"></td>
                <td th:text="${cliente.apellido}"></td>
                <td th:text="${cliente.email}"></td>
                <td th:text="${cliente.createAt}"></td>
            </tr>
        </tbody>
    </table>

</body>
</html>
```

Ahora en resources creamos el archivo import.sql con los datos de prueba.

insert into clientes(id,nombre,apellido,email,create_at) values
(1,'marcelino','torres','mtorres@unitru.edu.pe','2021-08-28');

insert into clientes(id,nombre,apellido,email,create_at) values
(2,'juan','perez','jperez@unitru.edu.pe','2022-03-20');

al Ejecutar sale:


La consola de H2

Nos vamos al archivo aplication.properties

```
spring.datasource.url=jdbc:h2:mem:clientesdb
spring.datasource.username=marcelino
spring.datasource.password=sa
spring.datasource.driver-class-name=org.h2.Driver
spring.h2.console.enabled=true
```

La base de datos se llama : testdb y el usuario es sa, sin clave

Levantamos el proyecto y escribimos : http://h2-console