

CRUD DE PERSONAS CON SPRING



Lección: Agregar una Persona con Spring Boot y Thymeleaf

● Introducción

En esta lección, implementaremos el caso de uso para **agregar un nuevo registro de persona** desde una aplicación web utilizando Spring Boot, Thymeleaf y una base de datos MySQL. Partimos desde un proyecto funcional con Spring Data JPA y conexión a base de datos configurada.



Paso 1: Agregar el enlace "Crear persona"

📁 Abrimos el archivo `src/main/resources/templates/index.html` Y agregamos el siguiente enlace para acceder al formulario de creación:

```
<a th:href="@{/agregar}">Crear persona</a>
```

Esto renderiza un botón/link en la página de inicio.

📌 Código completo `index.html`:

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
    <head>
        <title>Inicio</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <h1>Inicio</h1>
        <a th:href="@{/agregar}">Crear Persona</a>
        <h2>Listado de Personas</h2>
        <div th:if="${personas != null and !personas.empty}">
            <table border="1">
                <tr>
                    <th>Nombre</th>
                    <th>Apellido</th>
                    <th>Email</th>
                    <th>Telefono</th>
                </tr>
                <tr th:each="persona : ${personas}">
                    <td th:text="${persona.nombre}">Nombre</td>
                    <td th:text="${persona.apellido}">Apellido</td>
                    <td th:text="${persona.email}">Email</td>
                    <td th:text="${persona.telefono}">Telefono</td>
                </tr>
            </table>
        </div>
        <div th:if="${personas == null or personas.empty}">
            La lista de personas está vacía
        </div>
    </body>
</html>
```

📋 Paso 2: Crear el formulario en la nueva vista `modificar.html`

📁 Creamos un nuevo archivo HTML llamado `modificar.html` dentro de `src/main/resources/templates`.

El formulario usará **Thymeleaf** y se conectará al objeto `persona`.

Fragmento clave del formulario:

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org"
      xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>Datos de la Persona</title>
    <meta charset="UTF-8"/>
</head>
<body>
    <h1>Datos de la Persona</h1>
    <a th:href="@{/}">Regresar</a>
    <br/>
    <form th:action="@{/guardar}" method="post" th:object="${persona}">
        <br/>
        <label for="nombre">Nombre:</label>
        <input type="text" name="nombre" th:field="*{nombre}" />
        <br/>
        <label for="apellido">Apellido:</label>
        <input type="text" name="apellido" th:field="*{apellido}" />
        <br/>
        <label for="email">Email:</label>
        <input type="email" name="email" th:field="*{email}" />
        <br/>
        <label for="telefono">Teléfono:</label>
        <input type="tel" name="telefono" th:field="*{telefono}" />
        <br/>
        <input type="submit" name="guardar" value="Guardar"/>
    </form>
</body>
</html>
```

Paso 3: Crear el método GET /agregar en el controlador

 Abrimos `src/main/java/mx/com/gm/web/ControladorInicio.java`
Agregamos el siguiente método para renderizar la vista del formulario:

```
@GetMapping("/agregar")
public String agregar(Persona persona) {
    return "modificar";
}
```

Esto prepara un objeto `persona` vacío y muestra la vista `modificar.html`.



Paso 4: Crear el método POST /guardar en el controlador

En el mismo archivo `ControladorInicio.java`, agregamos:

```
@PostMapping("/guardar")
public String guardar(Persona persona) {
    personaService.guardar(persona);
    return "redirect:/";
}
```

Este método recibe los datos del formulario, guarda la persona en la base de datos usando el servicio, y redirige a la página principal.

📌 Código completo `ControladorInicio.java`:

```
package gm.web;

import gm.domain.Persona;
import gm.servicio.PersonaService;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;

@Controller
public class ControladorInicio {

    private static final Logger log = LoggerFactory.getLogger(ControladorInicio.class);

    @Autowired
    private PersonaService personaService;

    @GetMapping("/")
    public String inicio(Model model) {
        log.info("Ejecutando el controlador Spring MVC");
        var personas = personaService.listarPersonas();
        model.addAttribute("personas", personas);
        return "index";
    }
}
```

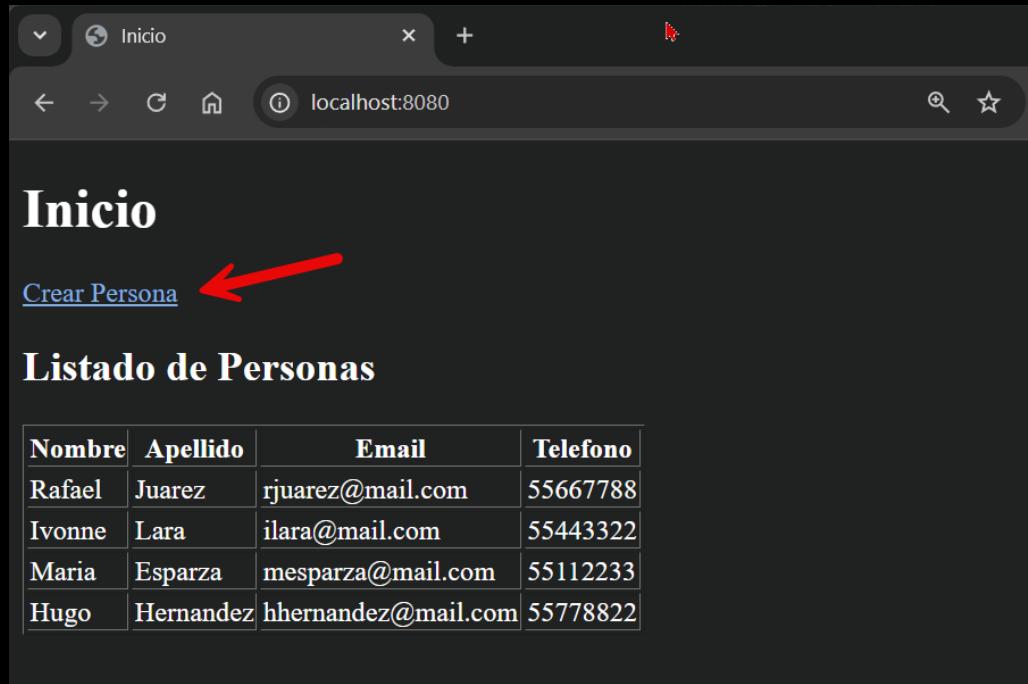
```
@GetMapping("/agregar")
public String agregar(Persona persona) {
    return "modificar";
}

@PostMapping("/guardar")
public String guardar(Persona persona) {
    personaService.guardar(persona);
    return "redirect:/";
}
}
```

💡 Paso 5: Probar funcionalidad

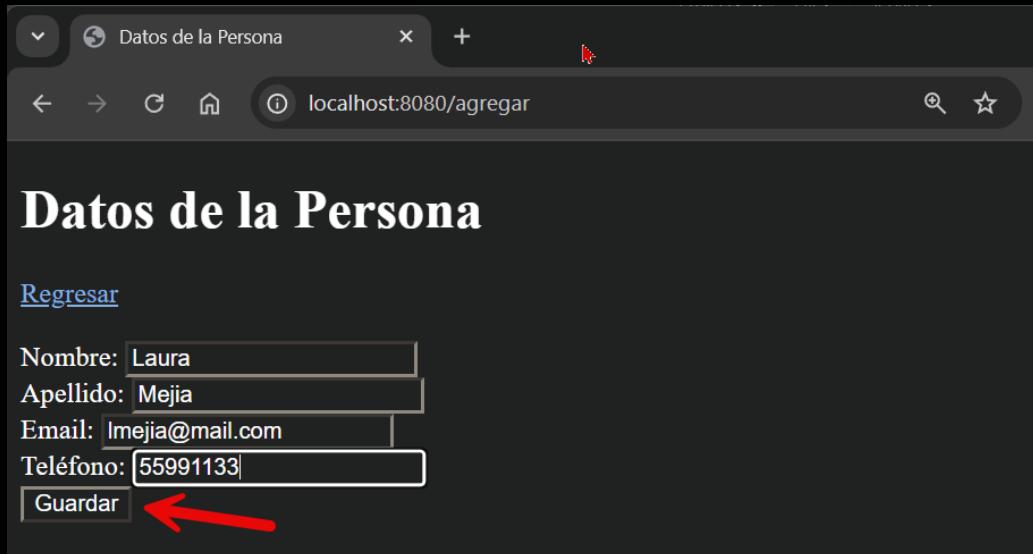
1. Inicia tu aplicación.
2. Abre <http://localhost:8080/>.
3. Da clic en "Crear persona".
4. Llena los campos y haz clic en "Guardar".

El nuevo registro se insertará en la tabla persona.



The screenshot shows a web browser window with the title 'Inicio'. Below the title, there is a button labeled 'Crear Persona' with a red arrow pointing to it. The main content area is titled 'Listado de Personas' and displays a table with the following data:

Nombre	Apellido	Email	Telefono
Rafael	Juarez	rjuarez@mail.com	55667788
Ivonne	Lara	ilara@mail.com	55443322
Maria	Esparza	mesparza@mail.com	55112233
Hugo	Hernandez	hhernandez@mail.com	55778822



Datos de la Persona

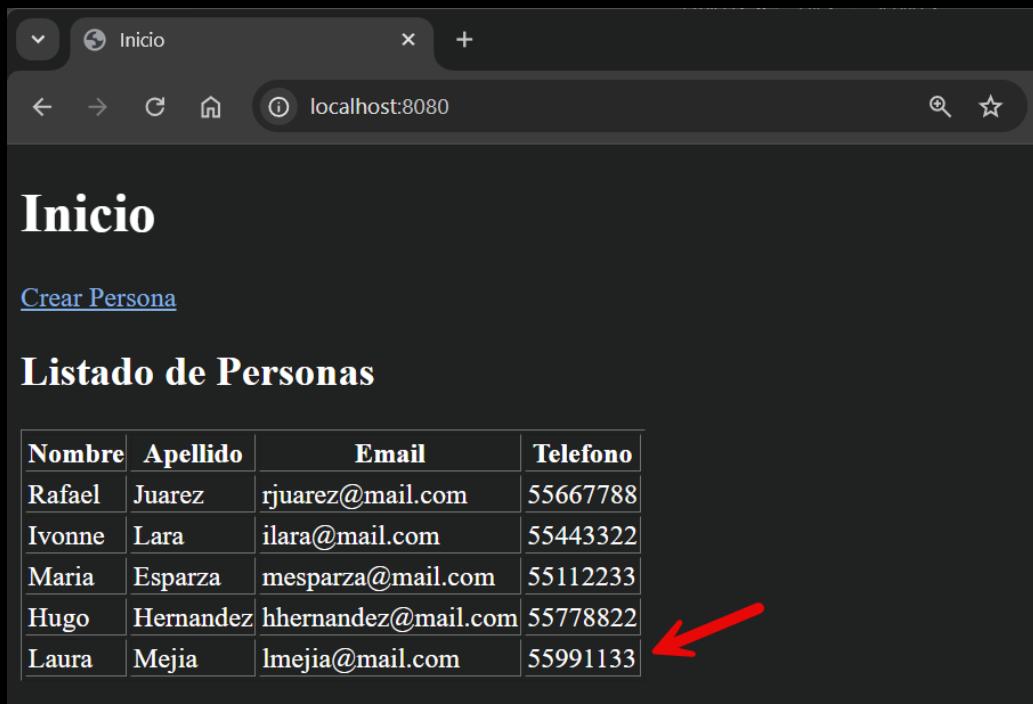
Regresar

Nombre:

Apellido:

Email:

Teléfono:



Inicio

[Crear Persona](#)

Listado de Personas

Nombre	Apellido	Email	Telefono
Rafael	Juarez	rjuarez@mail.com	55667788
Ivonne	Lara	ilara@mail.com	55443322
Maria	Esparza	mesparza@mail.com	55112233
Hugo	Hernandez	hhernandez@mail.com	55778822
Laura	Mejia	lmejia@mail.com	55991133

Conclusión

Con estos pasos, ya tenemos listo el caso de uso para **crear un nuevo registro de persona** en nuestra aplicación. Aprovechamos la inyección automática de dependencias y el manejo de formularios con Thymeleaf para lograrlo de forma sencilla y profesional.

Sigue adelante con tu aprendizaje  , ¡el esfuerzo vale la pena!

¡Saludos! 

Ing. Marcela Gamiño e Ing. Ubaldo Acosta

Fundadores de [GlobalMentoring.com.mx](#)