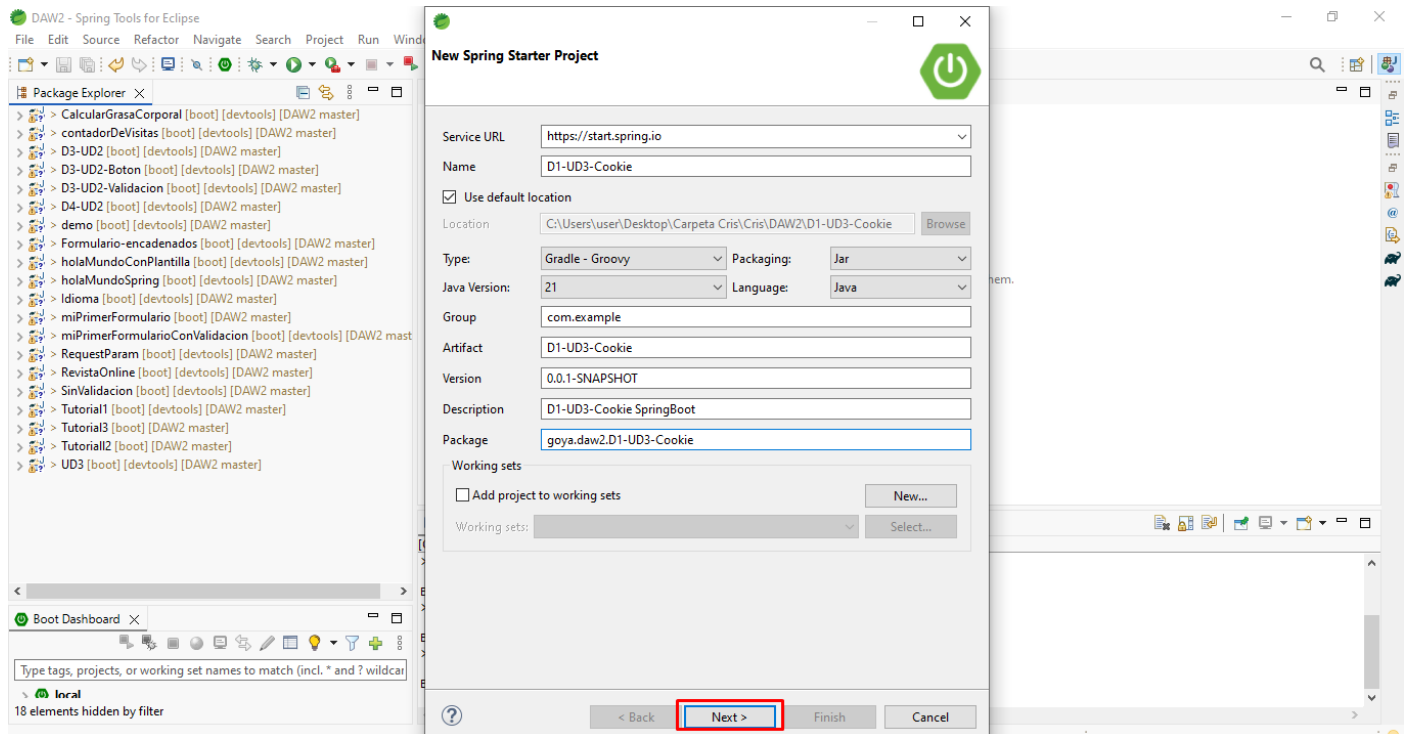


D1-UD3 enlazando formularios sin campos hidden

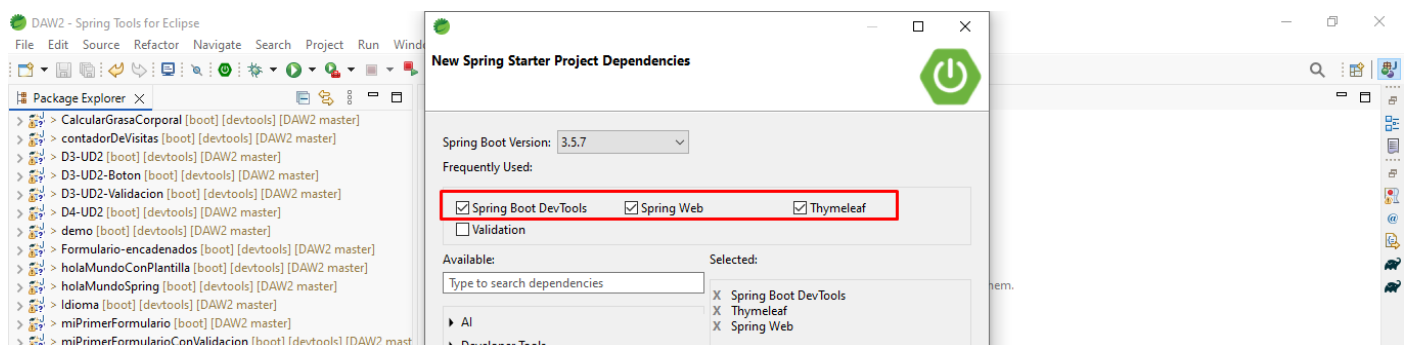
Se trata de modificar alguno de los retos previos o el adjunto para no usar campos ocultos (hidden, salvo para la etapa)

1. Con Cookies

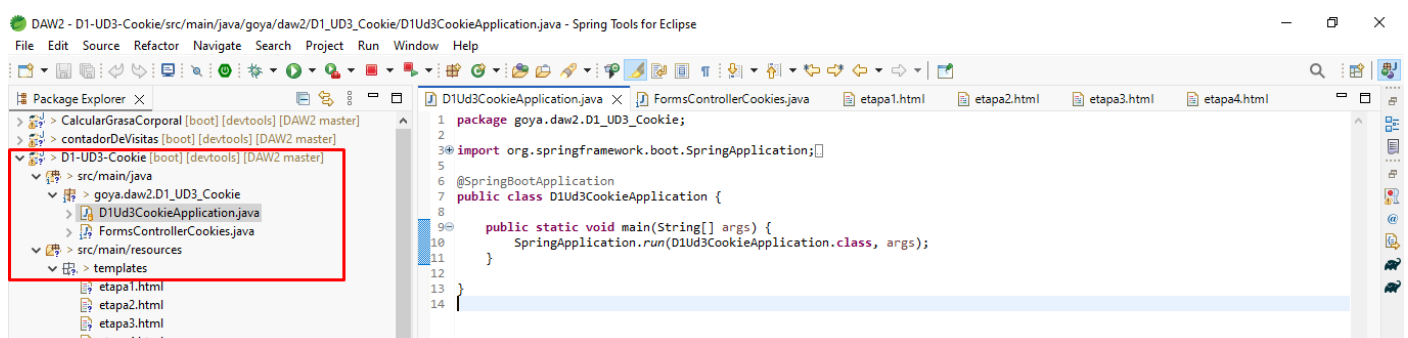
Nos creamos un proyecto llamado D1-UD3-Cookie



Añadimos las dependencias



Vemos que se ha creado el proyecto con la clase por defecto



Creamos la clase `FormsControllerCookies`, gestiona las etapas usando cookies para guardar y recuperar nombre, signo y aficciones, validando los datos en cada etapa. Finalmente muestra el resultado en la etapa 4 sin usar campos ocultos salvo `numEtapa`.

```

1 package goya.daw2.D1_UD3_Cookie;
2
3 import java.util.ArrayList;
4
5 @Controller
6 public class FormsControllerCookies {
7
8     static final String[] SIGNOS = { "", "Aries", "Tauro", "Géminis", "Cáncer", "Leo", "Virgo", "Libra", "Escorpio",
9         "Sagitario", "Capricornio", "Acuario", "Piscis" };
10
11     static final String[] AFICCIONES = { "Deportes", "Juega", "Lectura", "Relaciones sociales" };
12
13     @GetMapping("/cookies")
14     String getetapa1(@CookieValue(value = "nombre", required = false) String nombreCookie, Model model) {
15         model.addAttribute("numEtapa", 1);
16         if (nombreCookie != null) {
17             model.addAttribute("nombre", nombreCookie);
18         }
19         return "etapa1";
20     }
21
22     @PostMapping("/cookies")
23     String procesaetapa(@RequestParam(name = "numEtapa") Integer numEtapa,
24         @RequestParam(name = "nombre", required = false) String nombre,
25         @RequestParam(name = "signo", required = false) String signo,
26         @RequestParam(name = "aficciones", required = false) String[] aficciones,
27         @CookieValue(value = "nombre", required = false) String nombreCookie,
28         @CookieValue(value = "signo", required = false) String signoCookie,
29         @CookieValue(value = "aficciones", required = false) String aficcionesCookie, HttpServletResponse response,
30         Model model) {
31
32         modelo.addAttribute("signos", SIGNOS);
33         modelo.addAttribute("aficciones", AFICCIONES);
34
35         // Nombre
36         if (nombre == null || nombre.isBlank()) {
37             nombre = nombreCookie;
38         } else {
39             Cookie cookieNombre = new Cookie("nombre", nombre);
40             cookieNombre.setMaxAge(3600);
41         }
42
43         // Signo
44         if (signo == null) {
45             signo = signoCookie;
46         } else {
47             Cookie cookieSigno = new Cookie("signo", signo);
48             cookieSigno.setMaxAge(3600);
49             cookieSigno.setHttpOnly(true);
50             cookieSigno.setPath("/cookies");
51             response.addCookie(cookieSigno);
52         }
53
54         // Aficciones
55         if (aficciones == null) {
56             if (aficcionesCookie != null) {
57                 aficciones = aficcionesCookie.split(",");
58             }
59         } else {
60             Cookie cookieAficciones = new Cookie("aficciones", String.join(",", aficciones));
61             cookieAficciones.setMaxAge(3600);
62             cookieAficciones.setHttpOnly(true);
63             cookieAficciones.setPath("/cookies");
64             response.addCookie(cookieAficciones);
65         }
66
67         modelo.addAttribute("nombre", nombre);
68         modelo.addAttribute("signo", signo);
69         modelo.addAttribute("numEtapa", numEtapa);
70
71         // Validaciones
72     }
73 }

```

```

46 // Nombre
47 if (nombre == null || nombre.isBlank()) {
48     nombre = nombreCookie;
49 } else {
50     Cookie cookieNombre = new Cookie("nombre", nombre);
51     cookieNombre.setMaxAge(3600);
52     cookieNombre.setHttpOnly(true);
53     cookieNombre.setPath("/cookies");
54     response.addCookie(cookieNombre);
55 }
56
57 // Signo
58 if (signo == null) {
59     signo = signoCookie;
60 } else {
61     Cookie cookieSigno = new Cookie("signo", signo);
62     cookieSigno.setMaxAge(3600);
63     cookieSigno.setHttpOnly(true);
64     cookieSigno.setPath("/cookies");
65     response.addCookie(cookieSigno);
66 }
67
68 // Aficciones
69 if (aficciones == null) {
70     if (aficcionesCookie != null) {
71         aficciones = aficcionesCookie.split(",");
72     }
73 } else {
74     Cookie cookieAficciones = new Cookie("aficciones", String.join(",", aficciones));
75     cookieAficciones.setMaxAge(3600);
76     cookieAficciones.setHttpOnly(true);
77     cookieAficciones.setPath("/cookies");
78     response.addCookie(cookieAficciones);
79 }
80
81 modelo.addAttribute("nombre", nombre);
82 modelo.addAttribute("signo", signo);
83 modelo.addAttribute("numEtapa", numEtapa);
84
85 // Validaciones

```

Validaciones

```

80      modelo.addAttribute("nombre", nombre);
81      modelo.addAttribute("signo", signo);
82      modelo.addAttribute("numEtapa", numEtapa);
83
84      // Validaciones
85      String errores = "";
86      if (numEtapa == 1 && (nombre == null || nombre.isBlank())) {
87          errores = "Debes poner un nombre no vacío";
88      } else if (numEtapa == 1 && (nombre.length() < 3 || nombre.length() > 10)) {
89          errores = "La longitud del nombre debe estar entre 3 y 10";
90      }
91      if (numEtapa == 2 && (signo == null || signo.equals(""))) {
92          errores = "Debes seleccionar un signo";
93      }
94      if (numEtapa == 3 && (aficciones == null || aficciones.length == 0)) {
95          errores = "Debes elegir al menos una aficción, no seas soso/a";
96      }
97
98      if (!errores.isBlank()) {
99          modelo.addAttribute("errores", errores);
100         return "etapa" + numEtapa;
101     }
102
103     numEtapa++;
104     modelo.addAttribute("numEtapa", numEtapa);
105
106     if (numEtapa == 4) {
107         List<String> respuestas = new ArrayList<>();
108         respuestas.add(nombre);
109         respuestas.add(signos[Integer.parseInt(signo)]);
110         if (aficciones != null) {
111             aficciones = new String[] {};
112             respuestas.add(String.join(", ", aficciones));
113             modelo.addAttribute("respuestas", respuestas);
114         }
115         return "etapa" + numEtapa;
116     }
117 }
118
119

```

Creamos un HTML llamado etapa1.html, donde se pide al usuario que introduzca su nombre. Muestra errores si los hay y envía el formulario con numEtapa=1 para avanzar a la siguiente etapa

```

1 <!DOCTYPE HTML>
2 <html xmlns:th="http://www.thymeleaf.org">
3 <head><title>MiniQuiz etapa 1</title></head>
4 <body>
5 <form method="post" th:action="@{/cookies}">
6     <p th:text="${errores}"></p>
7     <label for="nombre">Nombre:</label>
8     <input type="text" name="nombre" th:value="${nombre}" />
9     <input type="hidden" name="numEtapa" value="1" />
10    <input type="submit" value="Siguiente" />
11 </form>
12 </body>
13 </html>
14

```

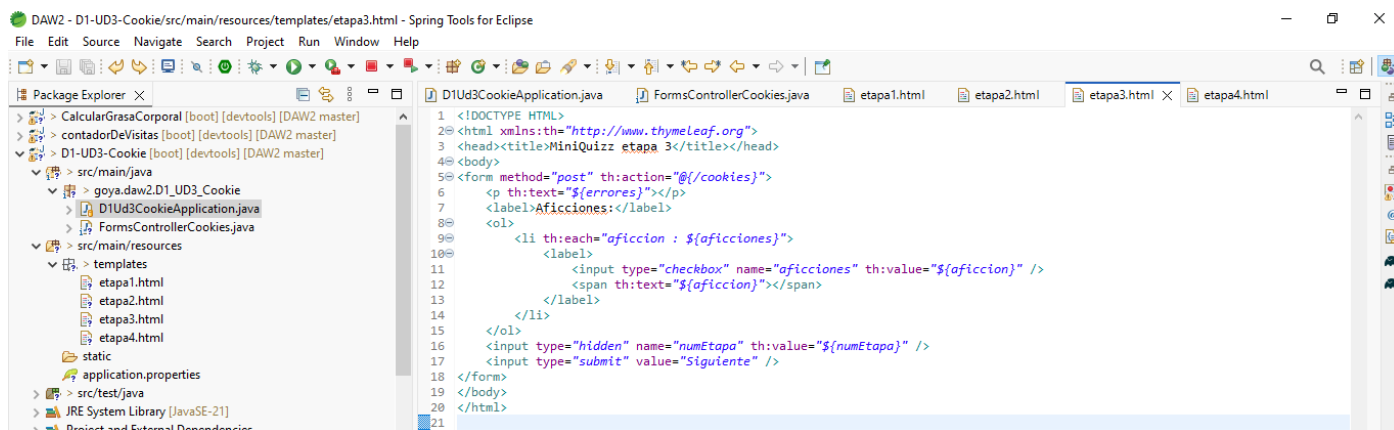
Creamos un HTML llamado etapa2.html, donde el usuario selecciona su signo del zodiaco. Muestra errores si los hay y envía el formulario con numEtapa=2 para pasar a la etapa siguiente

```

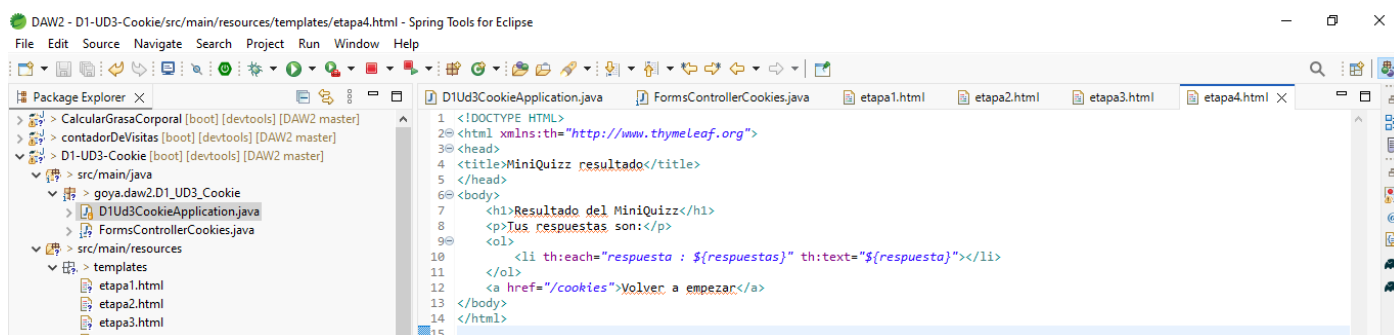
1 <!DOCTYPE HTML>
2 <html xmlns:th="http://www.thymeleaf.org">
3 <head><title>MiniQuiz etapa 2</title></head>
4 <body>
5 <form method="post" th:action="@{/cookies}">
6     <p th:text="${errores}"></p>
7     <label for="signo">Signo:</label>
8     <select name="signo">
9         <option th:each="signo, indice : ${signos}"
10             th:value="${signo.index}"
11             th:text="${signo}"></option>
12     </select>
13     <input type="hidden" name="numEtapa" th:value="${numEtapa}" />
14     <input type="submit" value="Siguiente" />
15 </form>
16 </body>
17 </html>
18

```

Creamos un HTML llamado etapa3.html, donde el usuario selecciona sus aficciones. Muestra errores si los hay y envía el formulario con numEtapa=3 para pasar a la etapa de resultados.



Creamos un HTML llamado etapa4.html, donde se muestran todas las respuestas en una lista ordenada, con un enlace para volver a empezar.



Navegador: <http://localhost:8080/cookies>

Lo primero que nos pide es el nombre

El signo y tenemos que elegir uno

Aficiones, es obligatorio poner uno

localhost:8080/cookies

Debes elegir al menos una aficción, no seas soso/a

Aficciones:

- ☐ Deportes
- ☐ Juerga
- ☐ Lectura
- ☐ Relaciones sociales

Siguiete

Resultado

localhost:8080/cookies

Resultado del MiniQuizz

Tus respuestas son:

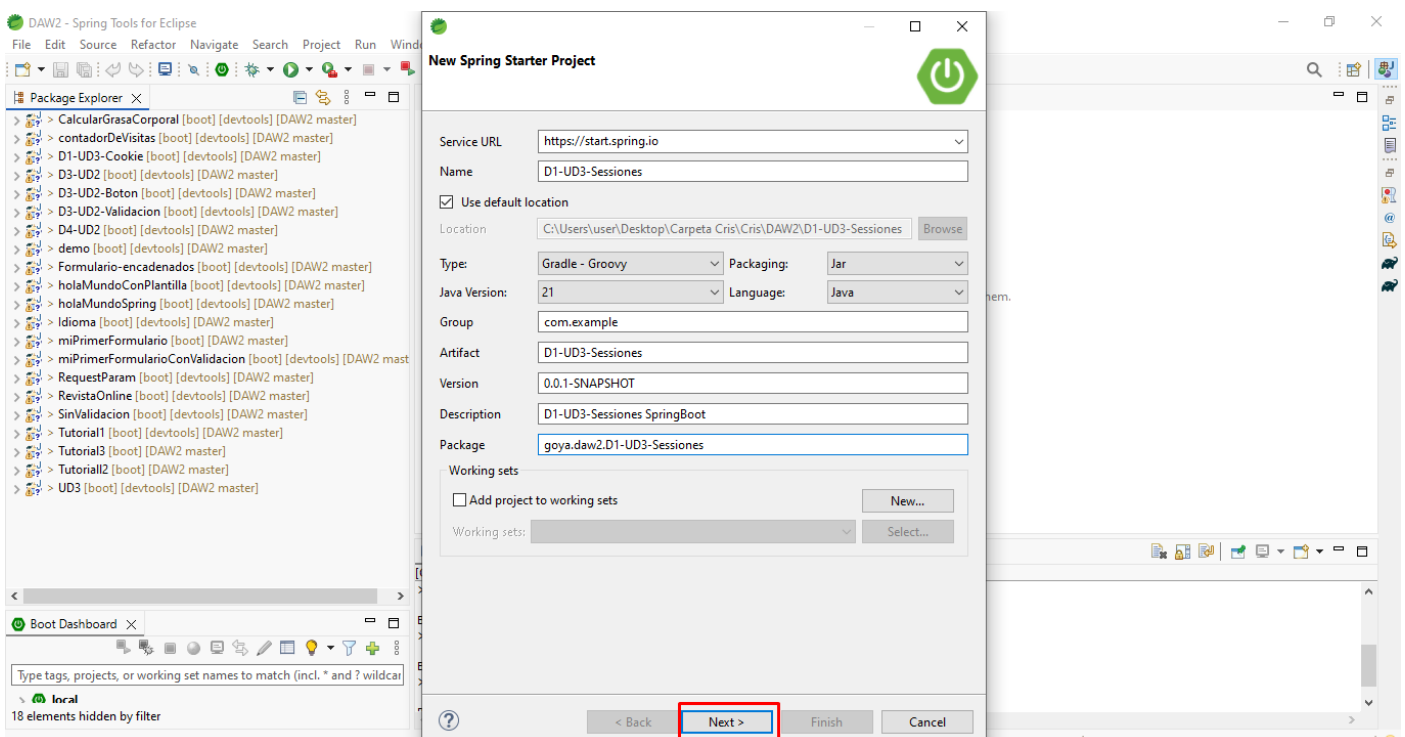
1. Cris
2. Virgo
3. Deportes

[Volver a empezar](#)

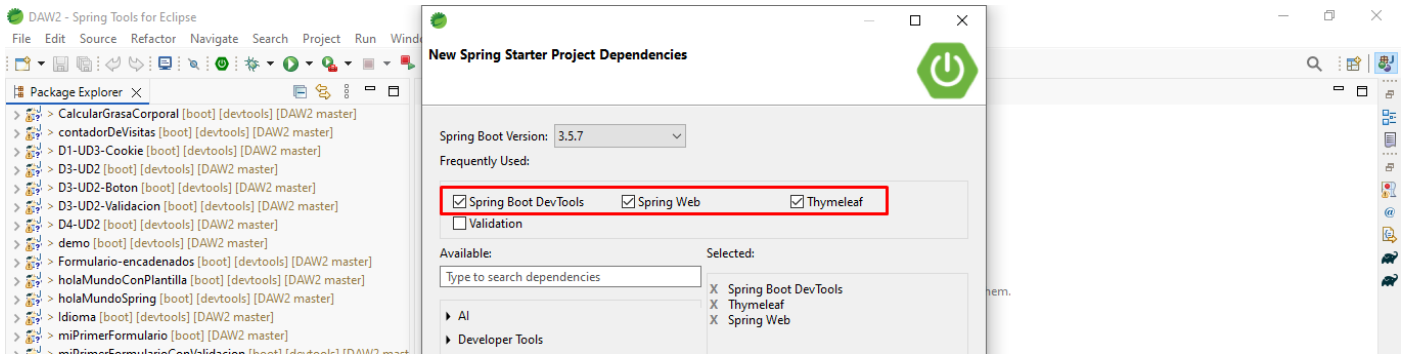
2. Con Sesiones

Nota: hay que hacerlo con RequestParams pues todavia no hemos visto como ligar un ModelAttribute con Cookie o Session. Si hiciste el reto previo con ModelAttribute (objeto) te sugiero usar el proyecto adjunto (importar como proyecto Gradle) o hacer uno nuevo desde cero.

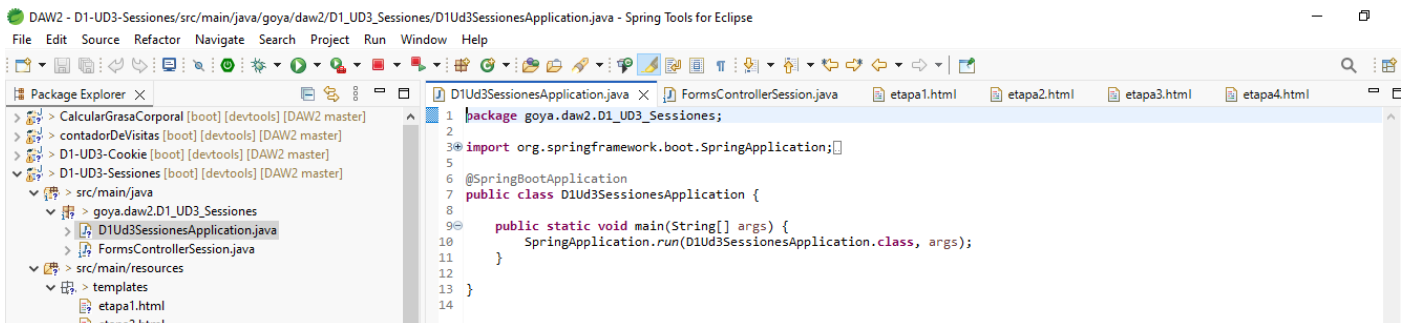
Nos creamos un proyecto llamado D1-UD3-Sesiones



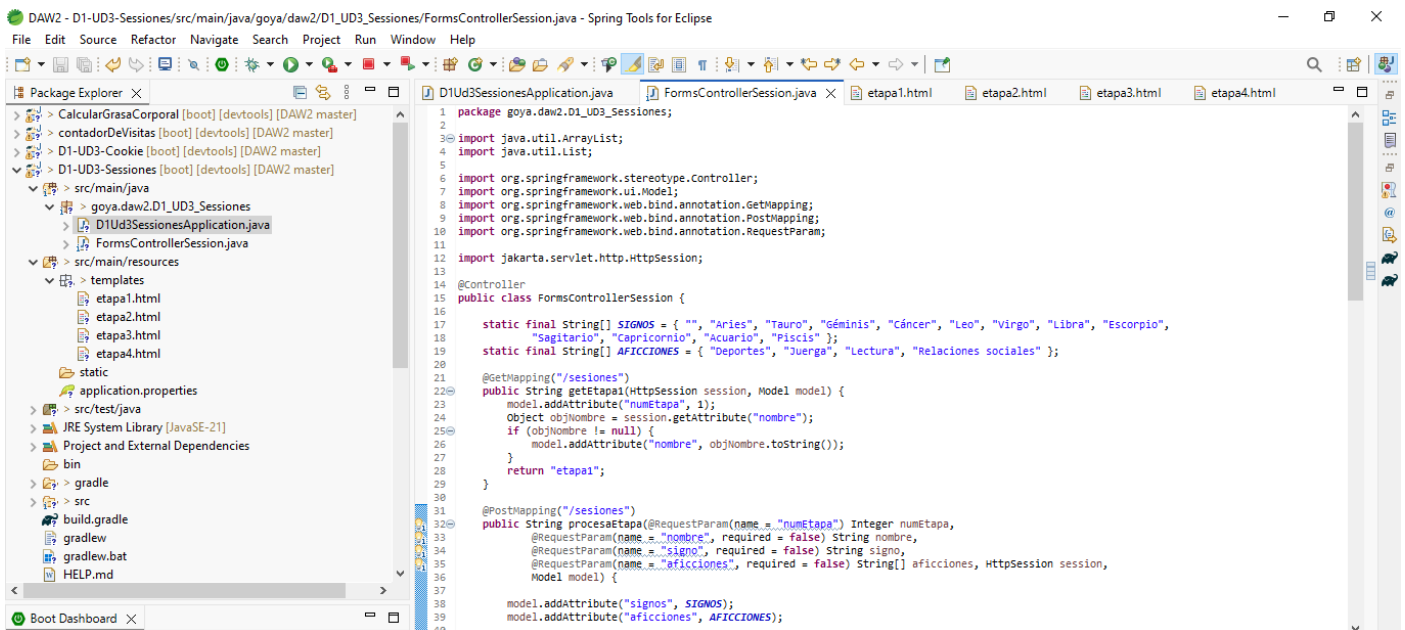
Añadimos las dependencias

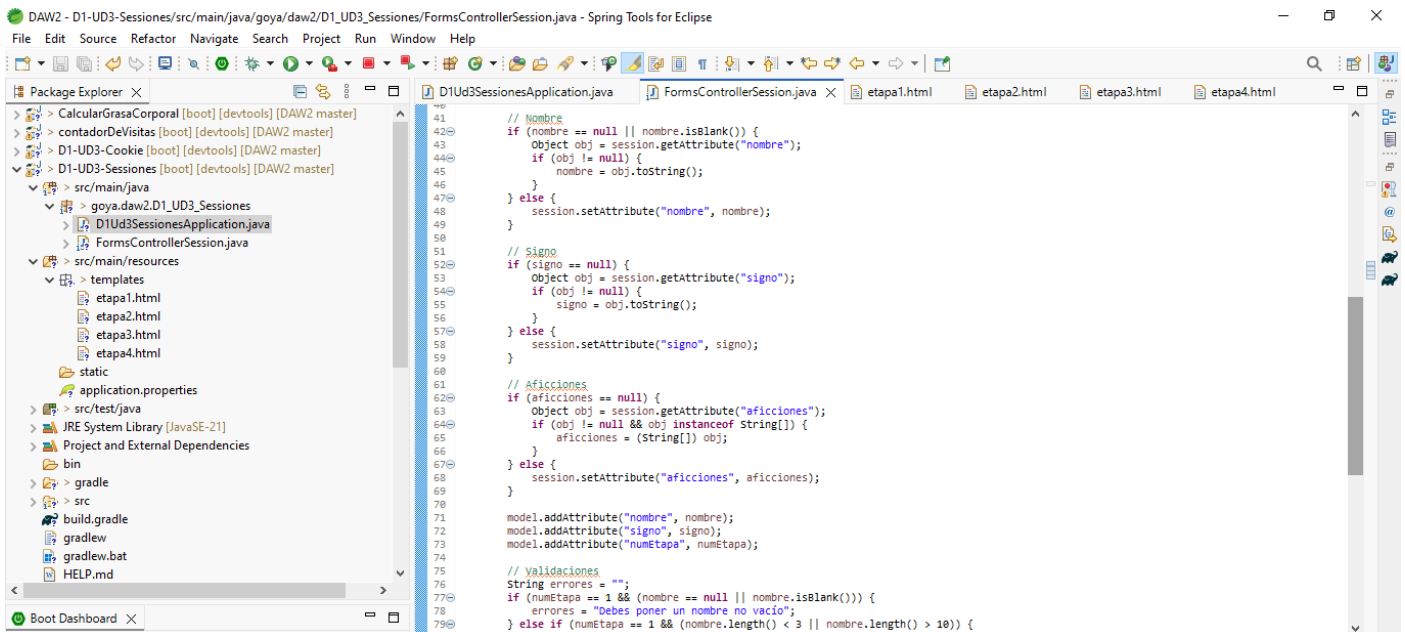


Vemos que se ha creado el proyecto y la clase por defecto

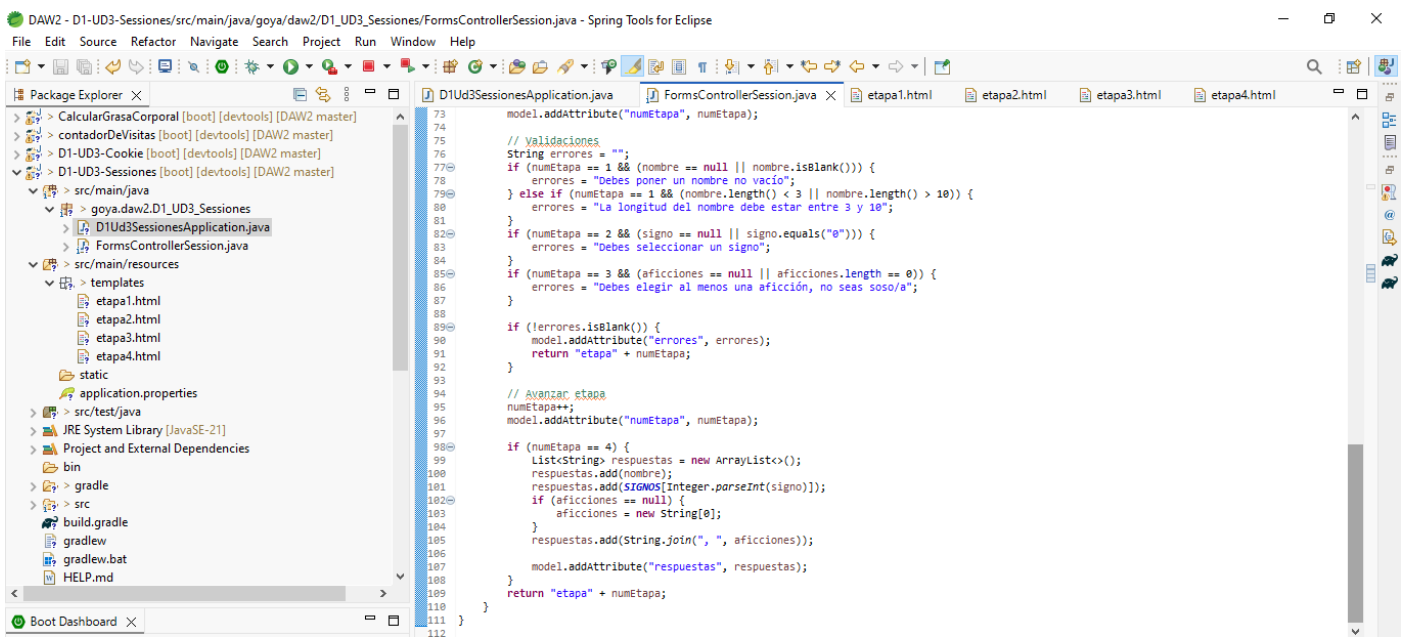


Creamos la clase FormsControllerSession, controla el formulario usando sesiones. Guarda y recupera los datos del usuario en la sesión del servidor, gestionando las distintas etapas del formulario y validando la información antes de mostrar el resultado final.

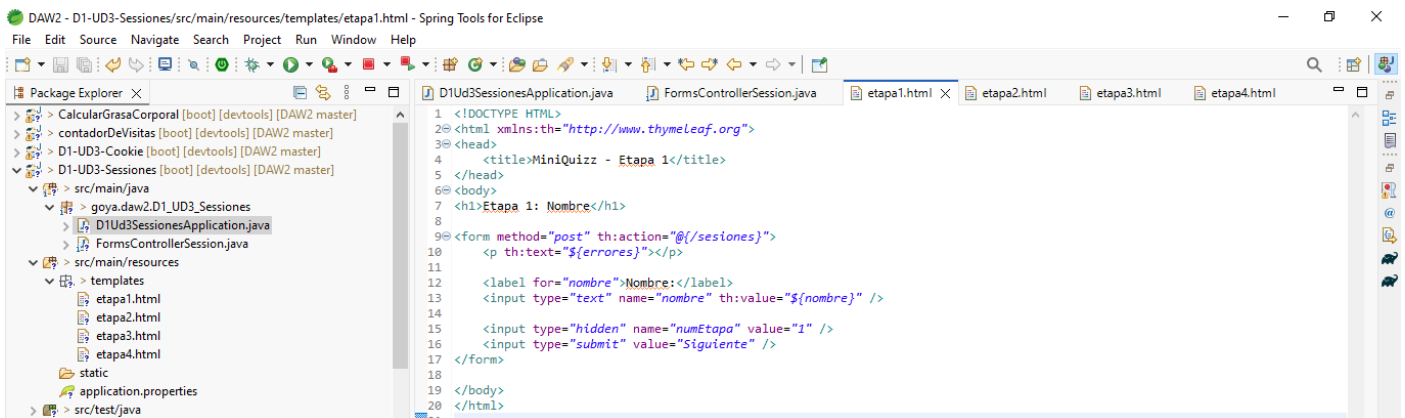




Validaciones

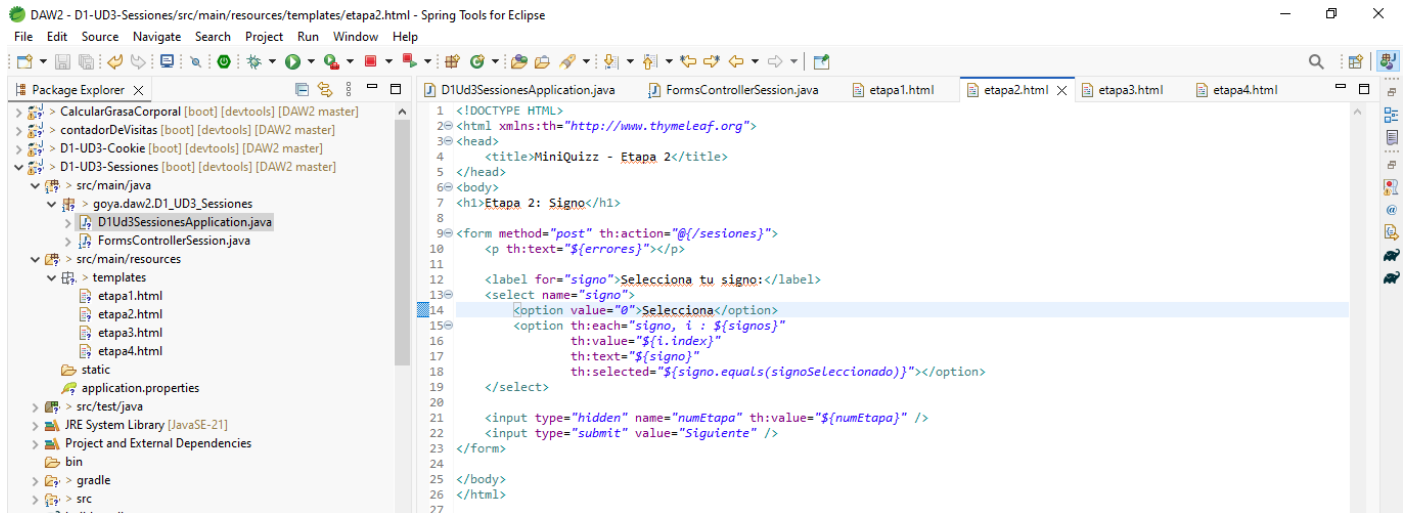


Creamos un HTML llamado etapa1.html, donde el usuario introduce su nombre. Incluye un campo de texto, un botón y si hay errores (por ejemplo: nombre vacío), los muestra en pantalla



Creamos un HTML llamado `etapa2.html`, donde el usuario selecciona su signo del zodiaco desde un menú desplegable.

También muestra posibles mensajes de error y tiene un botón para avanzar a la siguiente etapa.

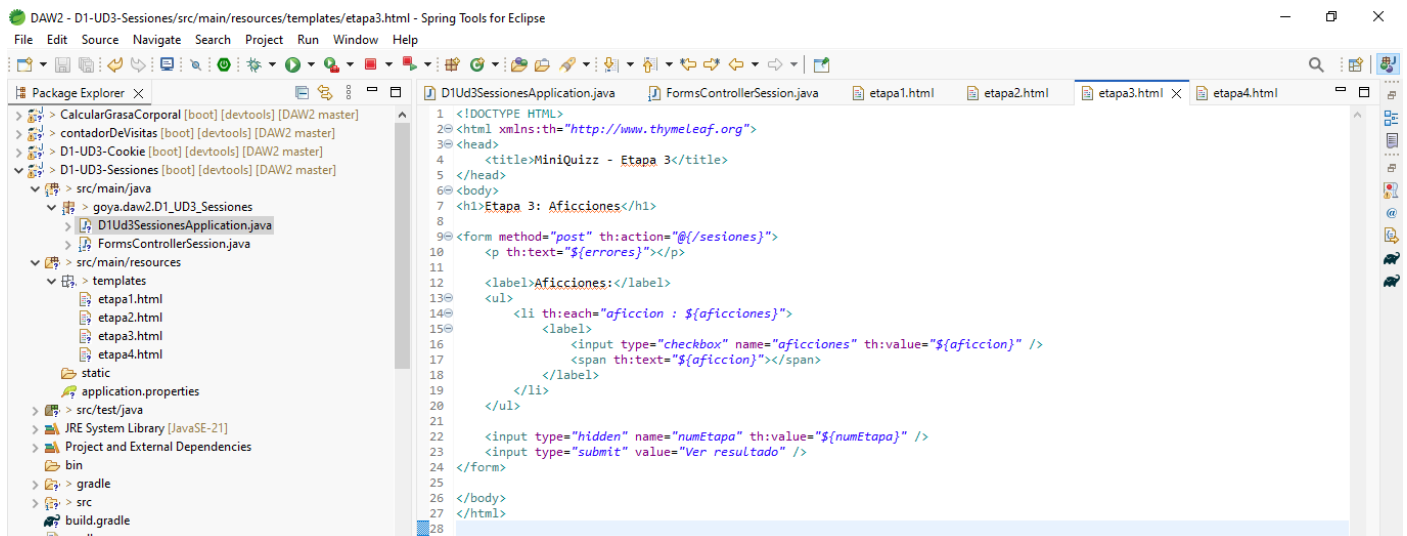


```

1 <!DOCTYPE HTML>
2 <html xmlns:th="http://www.thymeleaf.org">
3 <head>
4 <title>MiniQuiz - Etapa 2</title>
5 </head>
6 <body>
7 <h1>Etapa 2: Signo</h1>
8
9 <form method="post" th:action="@{/sesiones}">
10 <p th:text="${errores}"></p>
11
12 <label for="signo">Selecciona tu signo:</label>
13 <select name="signo">
14 <option value="0">Selecciona</option>
15 <option th:each="signo, i : ${signos}"
16 th:value="${i.index}"
17 th:text="${signo}"
18 th:selected="${signo.equals(signoSeleccionado)}"></option>
19 </select>
20
21 <input type="hidden" name="numEtapa" th:value="${numEtapa}" />
22 <input type="submit" value="Siguiente" />
23 </form>
24
25 </body>
26 </html>
27

```

Creamos un HTML llamado `etapa3.html`, donde el usuario elige sus aficciones mediante checkboxes. Permite seleccionar una o varias opciones y si no se elige ninguna, muestra un mensaje de error antes de avanzar a la etapa final.

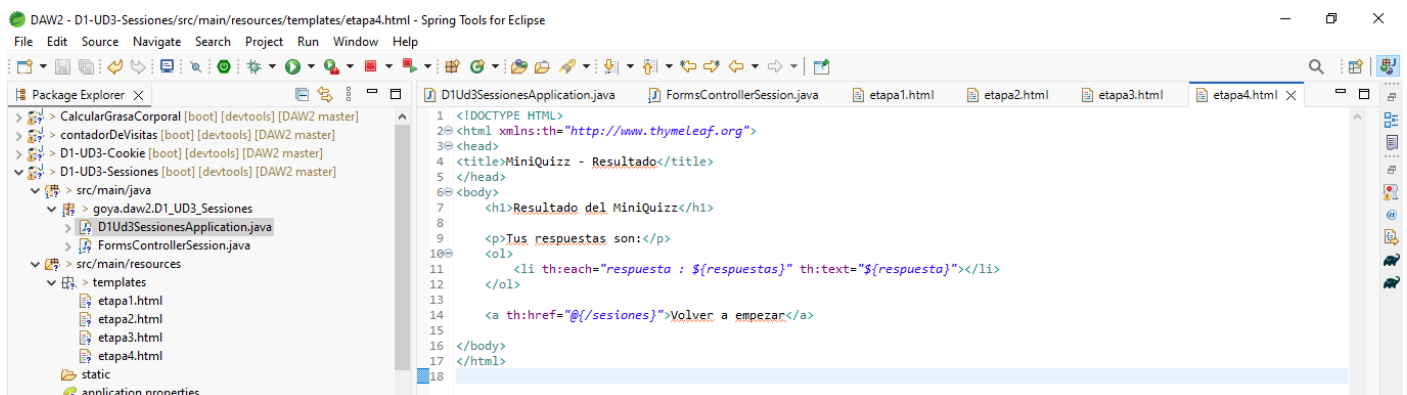


```

1 <!DOCTYPE HTML>
2 <html xmlns:th="http://www.thymeleaf.org">
3 <head>
4 <title>MiniQuiz - Etapa 3</title>
5 </head>
6 <body>
7 <h1>Etapa 3: Aficciones</h1>
8
9 <form method="post" th:action="@{/sesiones}">
10 <p th:text="${errores}"></p>
11
12 <label>Aficciones:</label>
13 <ul>
14 <li th:each="aficcion : ${aficciones}">
15 <label>
16 <input type="checkbox" name="aficciones" th:value="${aficcion}" />
17 <span th:text="${aficcion}"></span>
18 </li>
19 </ul>
20
21 <input type="hidden" name="numEtapa" th:value="${numEtapa}" />
22 <input type="submit" value="Ver resultado" />
23 </form>
24
25 </body>
26 </html>
27

```

Creamos un HTML llamado `etapa4.html`, donde se muestran las respuestas en una lista ordenada, junto con un enlace para volver a empezar el cuestionario desde el principio



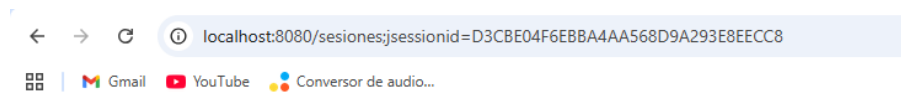
```

1 <!DOCTYPE HTML>
2 <html xmlns:th="http://www.thymeleaf.org">
3 <head>
4 <title>MiniQuiz - Resultado</title>
5 </head>
6 <body>
7 <h1>Resultado del MiniQuiz</h1>
8
9 <p>Tus respuestas son:</p>
10 <ol>
11 <li th:each="respuesta : ${respuestas}" th:text="${respuesta}"></li>
12 </ol>
13
14 <a th:href="@{/sesiones}">Volver a empezar</a>
15
16 </body>
17 </html>
18

```


Navegador: <http://localhost:8080/sesiones>

Ponemos nuestro nombre

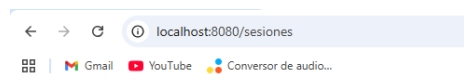


Etapa 1: Nombre

Debes poner un nombre no vacío

Nombre:

El signo



Etapa 2: Signo

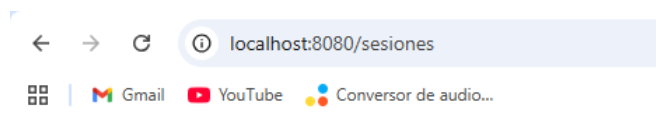
Debes seleccionar un signo

Selecciona tu signo:

Selecciona

Aries
Tauro
Géminis
Cáncer
Leo
Virgo
Libra
Escorpio
Sagitario
Capricornio
Acuario
Piscis

Aficiones



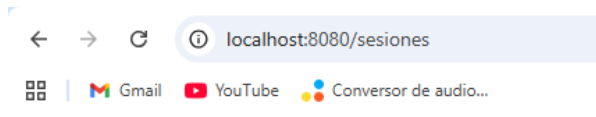
Etapa 3: Aficciones

Debes elegir al menos una aficción, no seas soso/a

Aficciones:

- ☐ Deportes
- ☐ Juerga
- ☐ Lectura
- ☐ Relaciones sociales

Resultado



Resultado del MiniQuizz

Tus respuestas son:

1. Cris
2. Cáncer
3. Juega, Lectura

[Volver a empezar](#)

Entrega: documento D1-UD3.PDF con la descripción de los pasos realizados, problemas encontrados, soluciones aportadas y conclusiones sobre la consecución de objetivos en cada apartado. El código se entregará en un enlace a cloud o git, ojo al .gitignore.

Evaluación:

- Funcionalidad correcta.
- Documentación clara y representativa del trabajo realizado.

Calificación:

- Suficiente: uno de los apartados funcional y documentado.
- Bien: los dos apartados funcionales y documentados.
- Excelente: lo anterior, pero con una documentación notable que incluya una reflexión sobre qué cambia en el ciclo de petición/respuesta y dónde se guardan los datos entre el apartado 1 y el 2 (no hace falta mencionar los cambios en el código)

Con cookies, los datos del usuario se guardan en el navegador del cliente y se envían al servidor con cada nueva petición HTTP.

Con sesiones los datos se almacenan en el servidor y el navegador solo guarda un identificador de sesión, el servidor puede recuperar la información del usuario sin reenviarla en cada petición.

Enlace GitHub:

<https://github.com/Cristina200317/PracticasEntornoServidor.git>