21 DE MAYO DE 2023

INFORME PRÁCTICA FINAL PAT

PROGRAMACIÓN DE APLICACIONES TELEMÁTICAS

SUSANA FRAGA, RUBÉN RODRÍGUEZ, MARÍA SARRADO

3ºB GITT

Introducción

Somos tres alumnos de ICAI que estudian doble grado de Ingeniería de Telecomunicaciones y Bussiness Analytics, y vamos a presentar nuestra práctica final de la asignatura de Programación de Aplicaciones Telemáticas. Este proyecto final consistía en crear una página web, empleando los distintos conocimientos aprendidos en clase.

La idea principal ha sido la siguiente: una página web orientada para ayudar a futuros estudiantes a encontrar la universidad en la que quieren estudiar, al igual que el alojamiento en tres ciudades distintas de España. Contamos con la opción de elegir entre cuatro universidades privadas en cada ciudad, según la especialidad que quiera estudiar el alumno, y luego en cuanto al alojamiento, se tiene la opción de elegir piso (del número de inquilinos de su gusto) o residencia (mixta, femenina o masculina).

A continuación, se presentarán cada una de las vistas para ver la funcionalidad que tienen y para saber cómo navegar entre ellas.

Vistas

Para empezar a explicar cada una de las vistas que tiene nuestra página web, primero debemos mencionar que tenemos un total de 10 vistas funcionales (con javascrpit) y ocho tablas. Para tener clara la estructura, estas son las tablas que tenemos:

- **Usuario**: id, email, credenciales, rol. Aquí se registrarán los emails (usuarios) que podrán acceder a nuestra página web iniciando sesión
- **Ciudad**: id, ciudad.
- **Colegio**: id, nombre, ciudad_id, tipo, descripción, url, web.
- **Piso**: id, nombre, ciudad_id, numero de inquilinos.
- **Universidad**: id, universidad, ciudad_id, url, web.
- **Carrera**: id, universidad_id, rama, carrera.
- **Contador**: id, nombre, valor.
- **Solicitud**: solicitud_id, usuario_id, universidad (de tipo Universidad), carrera (de tipo Carrera), tipo, colegio (de tipo Colegio), piso (de tipo Piso), estado.

INICIO (carpeta inicio en static)

En primer lugar, nada más arrancar el servidor, tenemos la página de *INICIO*.

En esta página explicamos en que consiste nuestra empresa y cuáles son las tres ciudades que ofrecemos: Madrid, Valencia y Barcelona. Al pinchar en cada uno de los nombres de las ciudades, te lleva directo a la página web del ayuntamiento de cada ciudad para conocer más en profundidad a que ciudad va a ir el estudiante.

Además, en la sección de “Procedimiento a seguir” tenemos lo siguiente. Las tres opciones que ofrecemos para poder solicitar (universidades, pisos, colegios mayores) que, si se pincha en cada uno de ellos, te redirige a la vista correspondiente con su resumen. Además, se explica que, para comenzar el proceso, te debes registrar o iniciar sesión. Para ello encontramos dos botones para iniciar sesión, si ya tenemos una cuenta, o registrarse si no disponemos de la cuenta en la página web.

Por último, en esta vista, se hace uso de javascript en la última sección, en donde contamos con estadísticas de nuestra página web. Es decir, permitimos ver cuantas personas han ido visitando

nuestra página web para ver el interés que se tiene. Para ello se ha hecho uso de una petición al endpoint de contador ***/api/contadores/{nombre}*** que es un @GetMapping del contador correspondiente.

UNIVERSIDADES (carpeta universidad en static)

Seguido de la página de inicio tenemos la página de las *UNIVERSIDADES*. En esta página, a parte de una breve introducción contamos con un desplegable que te permite seleccionar la ciudad en la que quieres estudiar. Al seleccionar la ciudad deseada, aparece una tabla de todas las universidades de esa ciudad en concreto. En la tabla, a parte del nombre de la universidad, tenemos la localización de la universidad en el mapa de la ciudad. No obstante, si interesa saber más de la universidad, incluimos el link de la página web oficial para que sea más fácil llegar a él.

Para todo ello, se ha hecho una petición al endpoint ***/api/{ciudadId}/universidad/*** de Universidad que es un @GetMapping que nos proporciona una lista de las universidades que pertenecen a esa ciudad correspondiente. Cambia cada vez que cambie el selector de la ciudad (***onchange***).

COLEGIOS MAYORES (carpeta colegios en static)

Seguido de la página de universidades, tenemos la página de *COLEGIOS MAYORES*. En esta página, también se tiene una pequeña introducción y, al igual que en universidades, contamos con un desplegable con el tipo de colegio mayor al que se quiere asistir: mixto, femenino o masculino. Al seleccionar el tipo que queremos, aparece una tabla con todos los colegios que pertenecen a ese tipo. Cada uno de los colegios mayores tiene su nombre, con una breve descripción del tipo de colegio que es y de los fundadores. También se proporciona una imagen para ver el colegio por fuera y un enlace al pinchar en el nombre del colegio para acceder a su página principal y conocerlo más a fondo.

Para todo ello, se ha hecho una petición al endpoint ***/api/{tipo}/colegio/*** de Colegio que es un @GetMapping que nos proporciona una lista de los colegios que pertenecen a ese tipo en concreto. Cambia cada vez que cambie el selector(***onchange***).

PISOS (carpeta pisos en static)

Seguido de la página de colegios mayores, tenemos la página de PISOS. En esta página, también se tiene una pequeña introducción y, al igual que en las anteriores, contamos con un desplegable con el número de inquilinos del piso que se quiere solicitar. Al seleccionar el número de inquilinos que queremos, aparece una tabla con todos los pisos que cumplen con esa característica.

Para todo ello, se ha hecho una petición al endpoint ***/api/{inquilinos}/piso/*** de Piso que es un @GetMapping que nos proporciona una lista de los pisos que cumplen con el número de inquilinos que se pasa. Cambia cada vez que cambie el selector(***onchange***).

REGISTRO (carpeta alta en static)

En esta vista, como dice su propio nombre, tenemos el registro del usuario que desea aplicar para el proceso. Se ha creado un formulario con nombre, apellidos, email y contraseña para poder registrar a ese usuario interesado. Dándole al botón de “Registrarse”, se hace una llamada al endpoint de usuario ***/api/usuarios*** que es un @PostMapping que hace un POST del nuevo usuario que se quiere registrar. En caso de que haya ya una cuenta, saltará un error diciendo que ya existe un usuario con ese mismo email.

En el caso de tener cuenta, aquí se da la opción de redirigirte a iniciar sesión.

COMENZAR PROCESO (carpeta sesión en static)

Teniendo ya una cuenta, en este caso, se inicia sesión con el email y las credenciales. Haciendo esto, se llama al endpoint de usuario ***/api/usuarios*** que se corresponde con un @GetMapping que hace un GET de ese usuario. En el caso de que fallen las credenciales, saltará un botón justo debajo que te dará la opción de volver a intentarlo. En el caso de que todo sea correcto, aparecerán debajo dos botones. Uno de ellos te deja salir de tu cuenta y otro es el que le permite al usuario comenzar todo el proceso de solicitud.

Un usuario va a poder hacer múltiples solicitudes, así que podrá repetir el proceso cuanto desee.

Si se quiere probar, hay un usuario ya creado con varias solicitudes cuyo email es **hola@gmail.com** y su contraseña es susana.

APLICACIÓN PRIMERO (carpeta aplicación_primero en static)

Una vez entramos en el proceso, lo primero que aparecen son dos selectores: Ciudad y Universidad. Hay que seleccionar una ciudad, y en función de la ciudad seleccionada, se rellenarán las casillas del selector de universidad. Cuando se escoja la universidad, aparecerá una tabla con las carreras que existen en esta universidad de esa ciudad. Esta tabla tiene la carrera junto con la rama a la que pertenece. El usuario deberá seleccionar solamente una casilla (checkbox) para seguir adelante con el proceso. Una vez lo tenga seleccionado, se pasará a la siguiente vista del proceso.

En este caso particular se ha hecho uso de los siguientes endpoints para poder llevarlo a cabo. En primer lugar, el endpoint de Universidad usado anteriormente ***/api/${ciudadId}/universidad/*** que devuelve la lista de universidades de esa ciudad. En segundo lugar, el endpoint de Carrera ***/api/universidad/{universidadId}/carrera*** que es un @GetMapping que devuelve la lista de carreras de la universidad seleccionada.

APLICACIÓN SEGUNDO (carpeta aplicación_segundo en static)

A continuación, se nos pasará a la selección del alojamiento. Solo se podrá en cada solicitud aplicar a o bien un piso o colegio mayor, de ahí que nuestro primer selector sea del tipo de alojamiento que se quiere. En función de lo seleccionado, aparecerá un nuevo selector en cada caso, el mismo que el usado en las páginas de Pisos y Colegios Mayores. La única diferencia es que, en este caso, se tiene en cuenta la ciudad seleccionada para que no puedas escoger alojamiento que no sea de la ciudad seleccionada.

Se han usado varios endpoints en ese caso, ya que, a parte de la funcionalidad anterior, se han realizado dos peticiones a dos endpoints que me proporcionan el nombre de la universidad y carrera anterior, que posteriormente se iba a usar en la solicitud, pero que al final no hizo falta. Se dejaron indicados para comprobar luego con los logs que era correcta la info. Estos dos endpoints son: ***/api/carrera/{carreraId}/*** y ***/api/universidad/{universidadId}/***, ya que tenemos guardados los valores de los id que se necesitan. Para la funcionalidad de la página en sí, se han usado otros dos, que en cada caso, devuelven una lista de los colegios o pisos según la ciudad y el tipo o número de inquilinos: ***/api/{colegioId}/{ciudadId}/colegio/*** y ***/api/{inquilinos}/{ciudadId}/piso/.***

APLICACIÓN ÚLTIMO (carpeta aplicación_ultimo2 en static)

Por último, se nos pasará a una página donde queremos que aparezca la solicitud que hemos seleccionado. Para que todo funcione correctamente, primero se ha de darle al botón "**ENVIAR SOLICITUD**" (un POST al siguiente endpoint de Solicitud ***/api/solicitudes*** que añade toda la info que

se necesita en la tabla de solicitud). Con este se hace el post correspondiente de la solicitud creada por el usuario. A continuación, se puede pulsar el botón de "**Muestra resumen**" (un GET al siguiente endpoint de Solicitud ***/api/solicitudes/{usuarioLogado.id}*** que muestra las solicitudes realizas por ese usuario) para que aparezca una tabla con tu solicitud. Por último, si se ha terminado de comprobar todo, se puede pulsar "**SALIR**" para que haya finalizado tu proceso.

CORRECCIONES

Finalmente, si hemos usado la práctica que nos has pasado y esto era lo que habíamos hecho mal y tu has corregido. Teníamos creados correctamente los endpoints para conseguir todas las variables, el problema era que nosotros, en vez de tener en el ***Entity*** de Solicitud variables de tipo Universidad, Colegio, etc, habíamos cogido directamente Longs para los ID de cada una de las cosas y Strings para los nombres. Para estas variables del Entity has usado la anotación @ManyToOne que significa que muchas solicitudes pueden tener la misma universidad, piso, carrera, etc.

Además, en el JAVASCRIPT has usado una notación específica para que, al asociar simplemente los ID de cada una de las cosas a su variable correspondiente, ya tenga esa variable toda su información:

```
universidad: { id: universidadId }

piso: pisoId ? {id: pisoId} : null,
```

Por ejemplo, al asociar de esta manera el id de la universidad, la variable definida como universidad en la solicitud, ya tendría todos sus valores. Con esto, por lo tanto, se pueden coger los valores que se quieran para la tabla (lo que nos interesan son los nombres) y que en la base de datos de solicitudes solo aparezcan los ID de cada cosa. En el caso del piso y colegio, se ha hecho de esa manera para que, si hay un nulo, en la tabla aparezca como un cuadro en blanco.

ESTADO SOLICITUD (carpeta estado en static)

En esta última vista, iniciando sesión de nuevo, se puede ver el estado de tu solicitud sin tener que volver a hacer el proceso para ver la tabla.

Conclusión

Como conclusión, hemos podido poner en práctica todo lo aprendido en clase haciendo un proyecto que podría ser perfectamente un problema de la vida real. Nos hemos encontrado con muchos problemas mientras realizábamos la página web, pero después de ir poco a poco probando distintas formas de hacerlo, hemos conseguido acabar la práctica de la mejor manera posible.

Los tests están en el controlador, dentro de MainApplicationTests. Están ahí explicados de cada una de las apis que se han comprobado.