

Prueba técnica

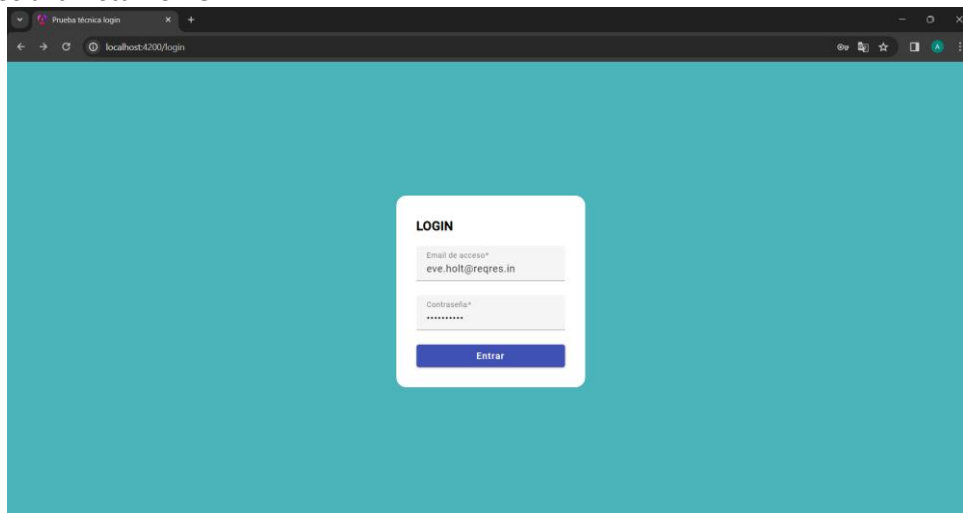
Introducción

La siguiente prueba se centra en evaluar las habilidades de programación de los candidatos, especialmente en el desarrollo de aplicaciones web utilizando Angular 17. La aplicación consistirá en un sencillo sistema de autenticación, manejo de rutas y manipulación de datos a través de la API <https://regres.in>.

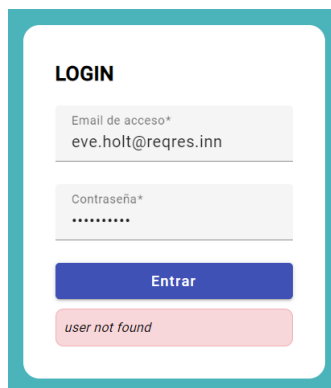
Descripción de la aplicación

Autenticación

Al acceder a la aplicación seremos redirigidos a una ventana de inicio de sesión donde iniciaremos sesión con el usuario eve.holt@regres.in. Debemos guardar el token proporcionado por la API y redirigirnos a la vista Home.



Al proporcionar un usuario incorrecto se nos indicará el error en la ventana.



Debemos controlar la validación del formulario, ambos campos deben ser obligatorios, de modo que, si alguno de los dos formularios se encuentra vacío el botón estará desactivado.

LOGIN

Email de acceso*

eve.holt@reqres.in

Contraseña*

Entrar













Home

Prueba técnica

Lista de usuarios

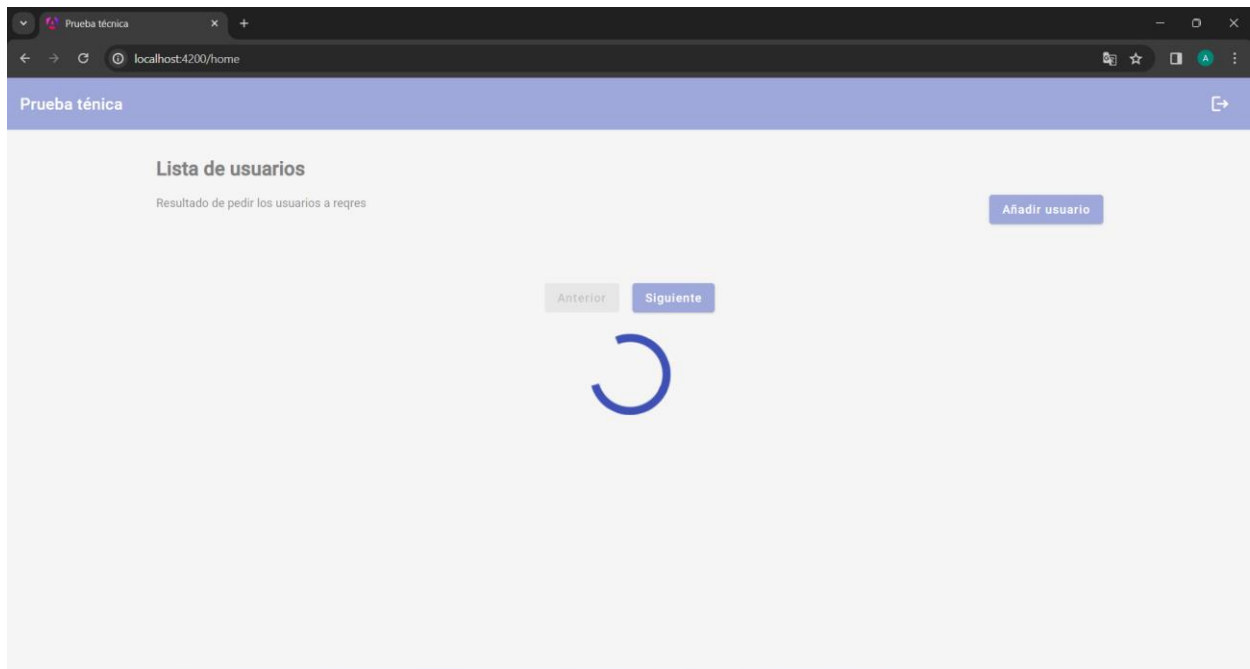
Resultado de pedir los usuarios a reqres

Añadir usuario

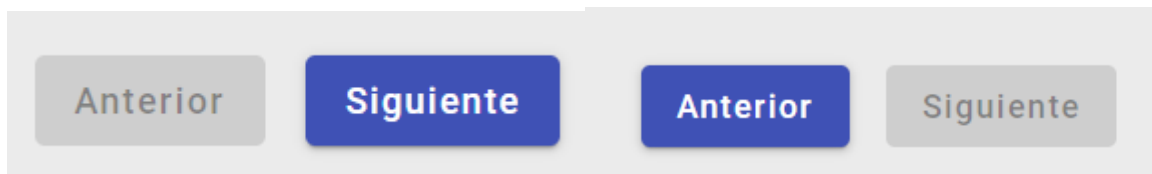
George Bluth george.bluth@reqres...	Janet Weaver janet.weaver@reqres...	Emma Wong emma.wong@reqres...	Eve Holt eve.holt@reqres.in	Charles Morris charles.morris@reqr...	Tracey Ramos tracey.ramos@reqres...
					
Más info 	Más info 	Más info 	Más info 	Más info 	Más info 

Anterior **Siguiente**

La ventana principal deberá recibir una lista de usuarios, los cuales deben mostrarse siguiendo un diseño responsivo. Para simular una llamada real, debemos implementar un delay, de modo que mientras recibe los datos, nuestra web mostrará un spinner, bloqueando así la manipulación de la web por parte del usuario.



Adicionalmente, debemos implementar una paginación, de modo que, al pulsar siguiente o anterior, se actualiza la lista de usuarios que mostramos. Si llegamos al final de la paginación, se bloqueará el botón de Siguiente, mientras que si nos encontramos en la primera página se bloqueará el botón de Anterior.

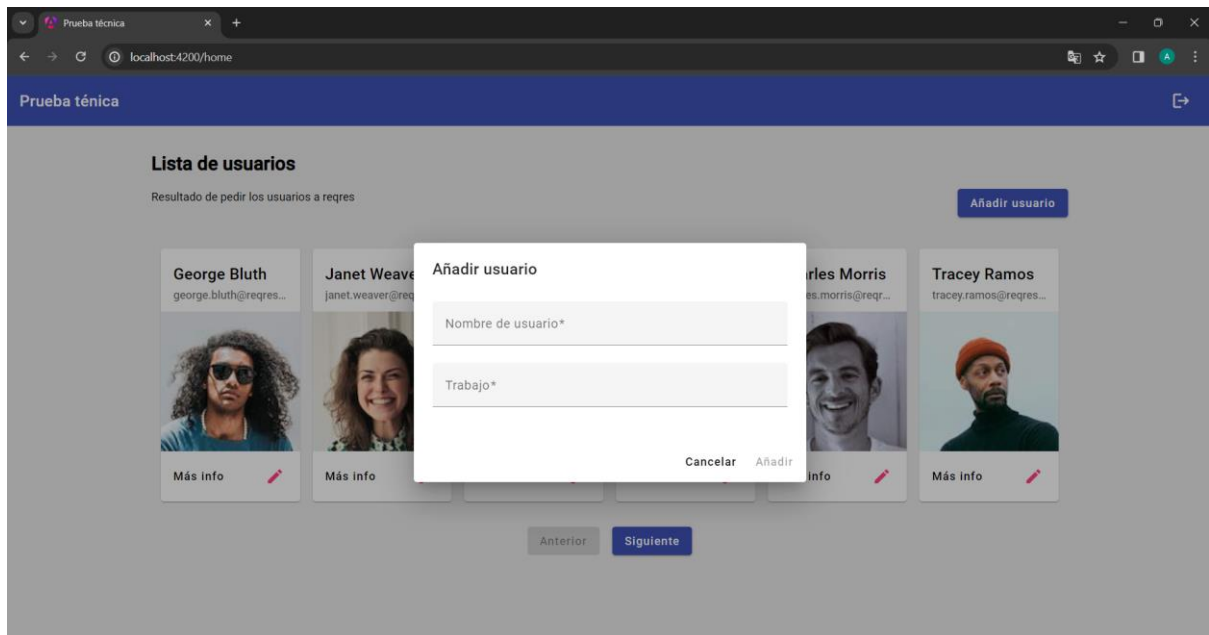


Nuestra ventana debe contar con los siguientes componentes:

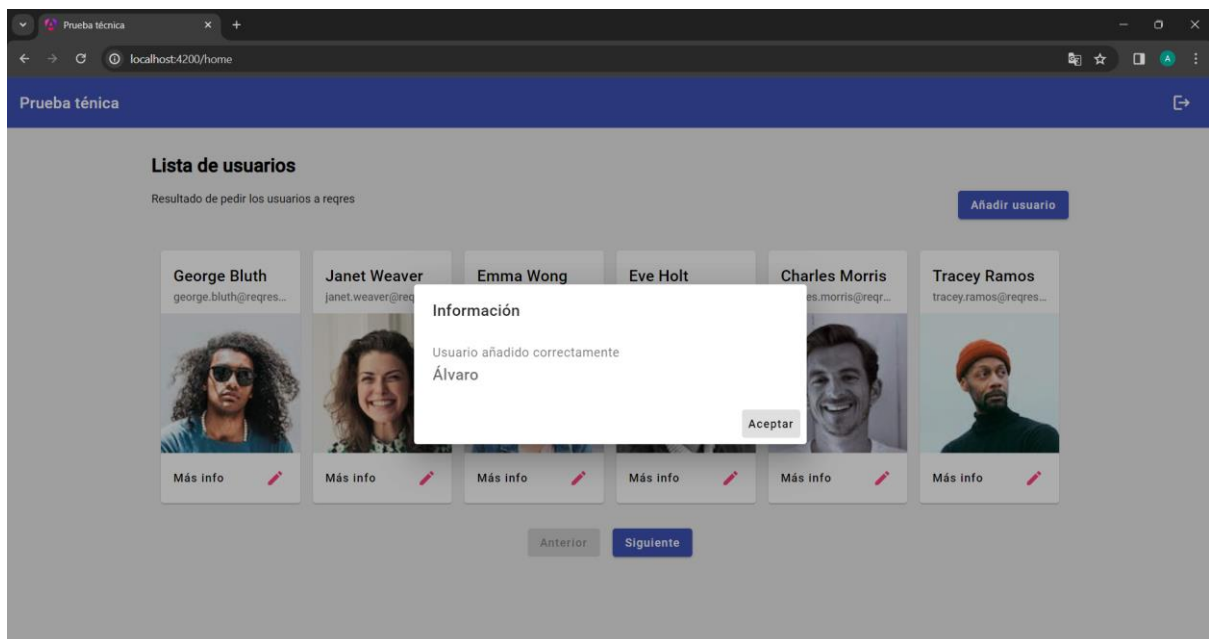
- Un header con un botón para poder cerrar sesión. Una vez cerrada la sesión seremos redirigidos a la vista de login.



- Un botón para añadir nuevos usuarios, el cual despliega un modal con un formulario para crear el usuario.



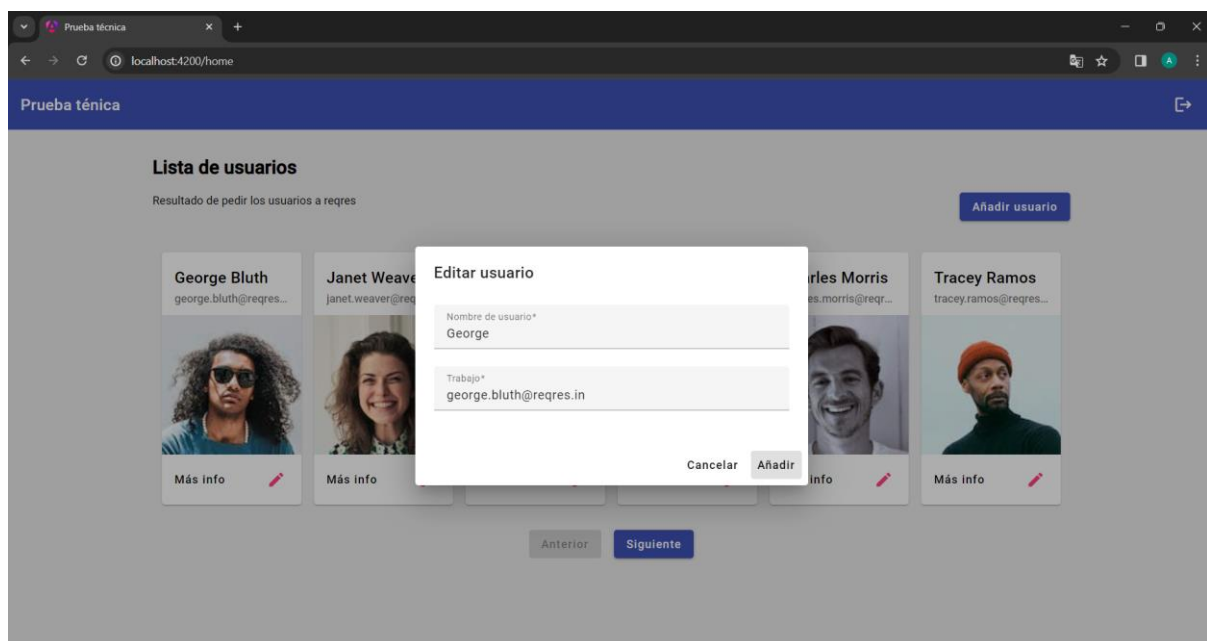
Al igual que el login, el botón debe estar desactivado si algún campo se encuentra vacío. Una vez enviamos el nuevo usuario, debemos mostrar el resultado que nos devuelve la API en otro modal.



-Dos botones por usuario recibido. Uno para mostrar más información en otra página y otro para editar la información.

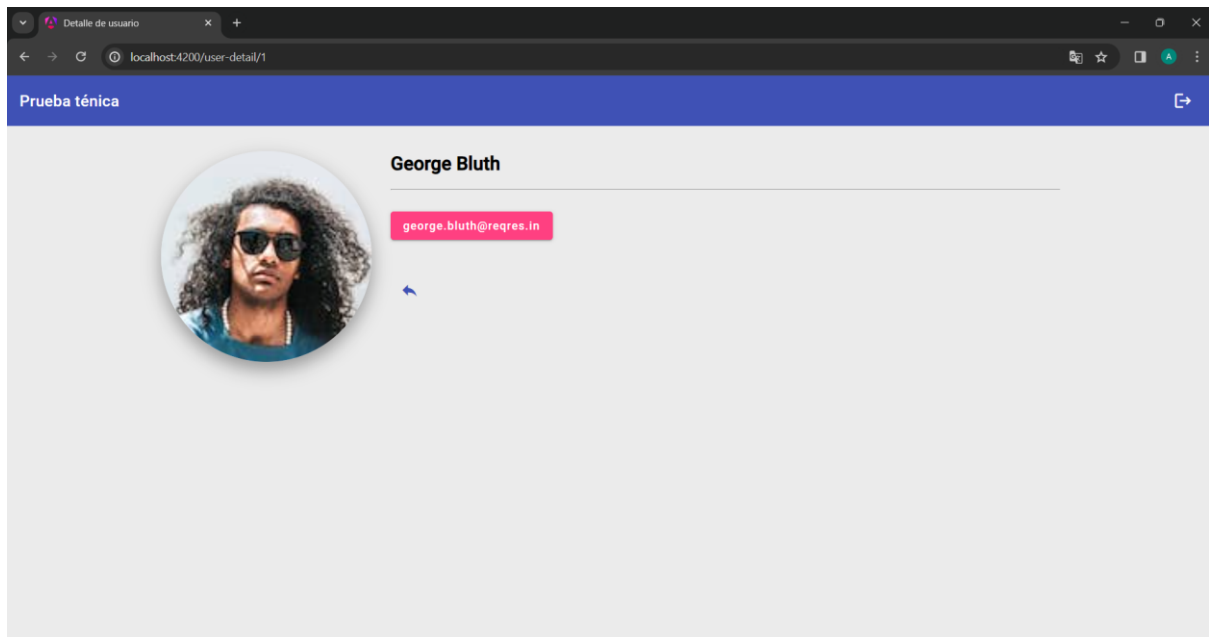


El botón de editar despliega el modal mostrado anteriormente con la información del usuario.

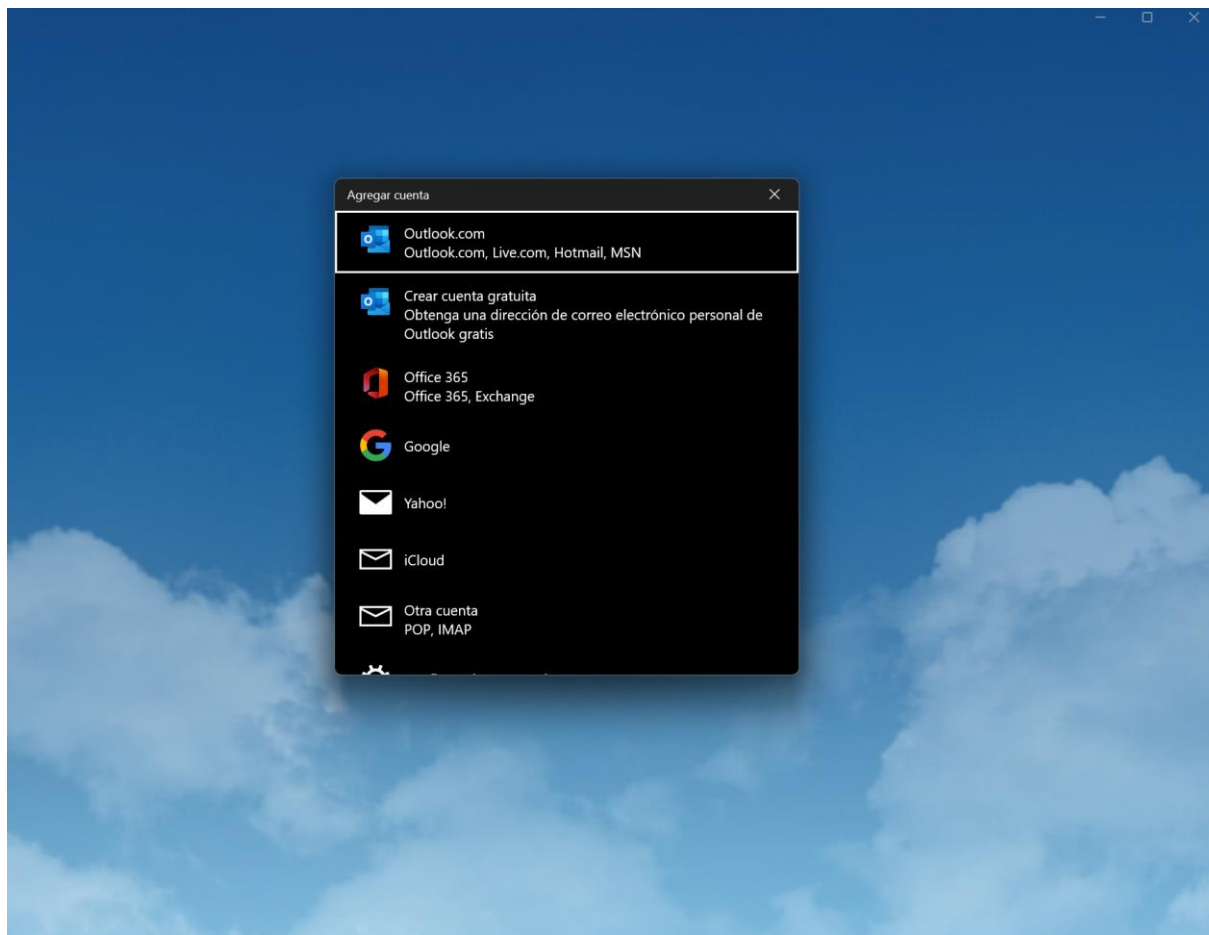


User-detail

Desde la vista Home, al pulsar el botón Más info seremos redirigidos a la vista que nos mostrará la información del usuario, pasando el id del usuario a la ruta.



La vista contará con un botón que permita enviar un email al usuario.



También tiene un botón para volver a la vista Home.

A valorar:

- Utilización de formularios reactivos
- Utilización de operadores RxJs
- Utilización de signals
- Utilización de interfaces
- Utilización del almacenamiento local para manipular el token de usuario
- Estructuración de la aplicación, limpieza de código y documentación

Para nota:

- Transiciones suaves entre las vistas
- Diseño responsive completo
- Loading (Spinner Component) mientras carga la información.
- Si la información no cabe en la caja de los usuarios, sustituir por puntos suspensivos