



Ejercicio Variables de Sesión

[Evaluación del ejercicio](#)

Vamos a reutilizar el archivo email-password.html que hemos usado para el sistema de autenticación de usuarios con Firebase.

A screenshot of the Firebase Authentication interface. At the top is a blue header with the text "Firebase Authentication". Below this is a white box with a blue header "Firebase Email & Password Authentication". Inside the white box, there is a prompt: "Enter an email and password below and either sign in to an existing account or sign up". Below the prompt are two input fields labeled "Email" and "Password". Under the "Email" field are two buttons: "SIGN IN" and "SIGN UP". Under the "Password" field are two buttons: "SEND EMAIL VERIFICATION" and "SEND PASSWORD RESET EMAIL". At the bottom of the white box, it says "Firebase sign-in status: Signed out" and "Firebase auth currentUser object value: null".

Si conseguimos autenticarnos, seremos redireccionados a la página que usamos como ejemplo en el tema de Express (por ejemplo, /users).

Nota: la sesión de Express y la sesión de Firebase NO son la misma y debemos gestionarlas de forma separada.

Con una cuenta con email "adrian@gmail.com" (tiene que estar previamente registrada en nuestra aplicación de Firebase) y la contraseña correcta, he sido redireccionado así a /users:

Hello adrian@gmail.com

Insertar cliente

Nombre
Apellido
Email

Customers

- Jane Doe - [Edit](#) - [Delete](#)
- Bob Smith - [Edit](#) - [Delete](#)
- Jill Jackson - [Edit](#) - [Delete](#)
- Adrian Nunez - [Edit](#) - [Delete](#)

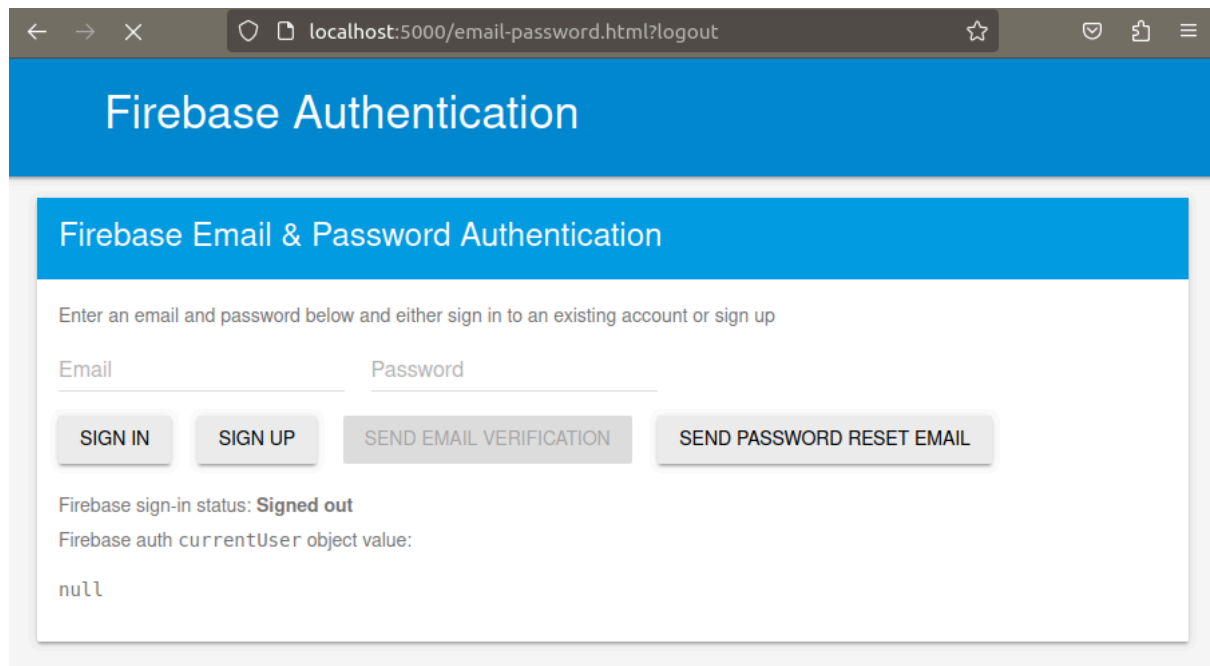
[Logout](#)

La página tendrá escrito un saludo en el que incluya el email de la sesión. Además, como estamos autenticados, aunque volvamos a / seremos redireccionados de vuelta a /users.

En esta página /users tendremos disponible el formulario para incluir usuarios de la entrega de Express (con las opciones de añadir, eliminar y editar clientes). Todas tienen que funcionar correctamente.

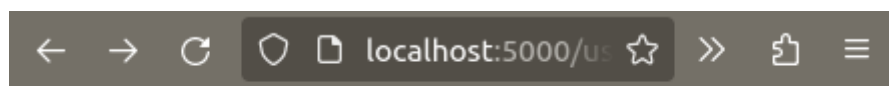
Además, la sesión de Express debe de estar guardada en MongoDB, no en memoria.

También se incluye un enlace “Logout” para ser redireccionados a / y destruir la sesión. En la URL que le pasamos para volver a / le pasaremos un parámetro para identificar que hemos cerrado la sesión (?logout en la URL):



Modificaremos email-password.html para tener en cuenta este parámetro y hacer el sign out. Es decir, si la URL incluye este parámetro, se hace un sign out automáticamente. Para ello, hay que comprobar la URL actual y ver si incluye “?logout”.

Por último, si intentamos acceder a /users si estar autenticados debería de aparecer un mensaje diciendo que tenemos que autenticarnos primero.



Please login first.

Desde el droplet, para lanzar el servidor usaremos este comando:

```
GOOGLE_APPLICATION_CREDENTIALS='ruta_fichero_json_descargado/nombre_del_fichero.json' npm start
```

Evaluación

Es necesario

- entregar los ficheros en esta entrega además de la IP+puerto. Si sólo se entrega uno de los dos o ninguno, la nota será de 0.
- crear un usuario en Firebase con email “dawe@gmail.com” y contraseña “123456”.

La puntuación será de 0 en caso de que el proyecto no esté dividido en carpetas correctamente. En concreto, se espera la siguiente estructura (no todos los elementos son necesarios):

- public/
 - |_____ - index.html (si hubiese)
 - js/
 - |_____ - main.js
- views/ (si hubiese)
 - |_____ - partials/ (si hubiese)
 - xxx.ejs (si hubiese)
- imagenes/ (si hubiese)
- server.js (o como lo llaméis)
- package.json

Tarea	Puntuación (sobre 10)
Autenticación y redireccionamiento a /users	3
Aviso de no estar autenticado en /users	1
La aplicación de gestión de usuarios de /users funciona correctamente	2
No nos permite volver a / si no hacemos logout	1
Logout (destrucción de sesión, redirección a / y cierre de la sesión en Firebase)	3