

Login en Angular – parte 2

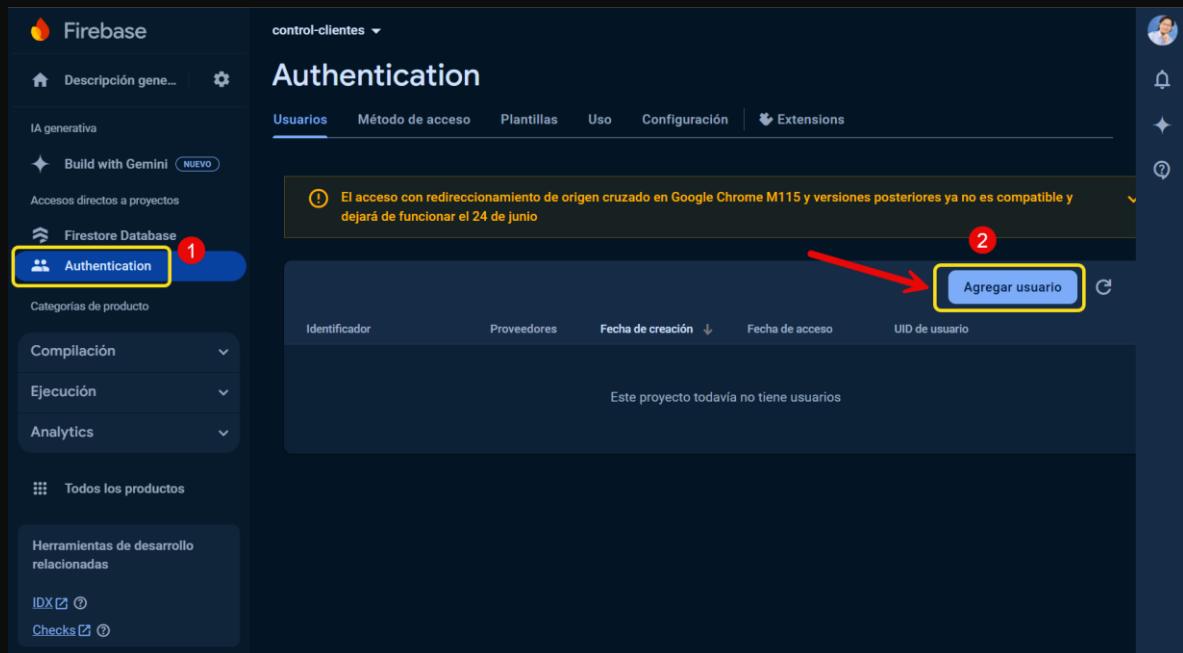


Login en Angular – Parte 2

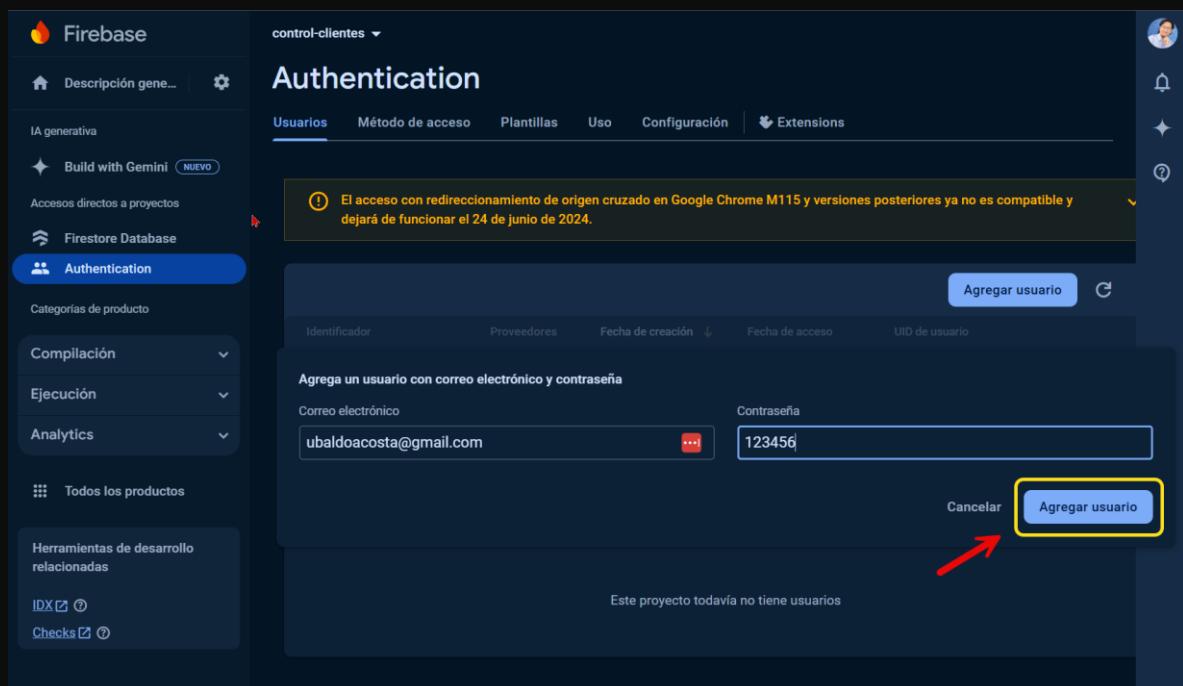
En esta lección, aprenderemos cómo crear el servicio de login para autenticarnos en nuestra base de datos de **Cloud Firestore** utilizando Firebase. También veremos cómo manejar los mensajes de error durante el proceso de autenticación.

Paso 1: Crear un Usuario Válido

Antes de implementar el servicio de login, asegúrate de que hayas agregado un usuario válido en Firebase Authentication para probar la funcionalidad de login en nuestra aplicación.



The screenshot shows the Firebase Authentication section. A red arrow points from the number 1 to the 'Authentication' tab in the sidebar. Another red arrow points from the number 2 to the 'Agregar usuario' (Add user) button in the main panel.



The screenshot shows the 'Agregar usuario' (Add user) dialog box. It has fields for 'Identificador' (Identifier), 'Proveedores' (Providers), 'Fecha de creación' (Creation date), 'Fecha de acceso' (Access date), and 'UID de usuario' (User ID). Below these fields, there is a note: 'Agrega un usuario con correo electrónico y contraseña' (Add a user with email and password). The 'Correo electrónico' (Email) field contains 'ubaldoacosta@gmail.com' and the 'Contraseña' (Password) field contains '123456'. A red arrow points from the 'Agregar usuario' button at the bottom right of the dialog to the number 3.

Paso 2: Crear el Servicio de Login

Usamos el siguiente comando para generar el servicio de login:

```
ng g s servicios/login --skip-tests
```

Código archivo login.service.ts:

```

import { Injectable } from '@angular/core';
import { Auth, signInWithEmailAndPassword } from '@angular/fire/auth';

@Injectable({
  providedIn: 'root' // Mantiene el servicio como inyectable en toda la aplicación
})
export class LoginService {
  // Inyectamos el servicio Auth en el constructor
  constructor(private authService: Auth) {}

  login(email: string, password: string) {
    return new Promise((resolve, reject) => {
      signInWithEmailAndPassword(this.authService, email, password)
        .then(datos => resolve(datos))
        .catch(error => reject(error));
    });
  }
}

```

Explicación:

- El servicio `LoginService` inyecta el servicio de autenticación de Firebase (`Auth`).
- El método `login` utiliza `signInWithEmailAndPassword` para intentar iniciar sesión con las credenciales proporcionadas. Si el inicio de sesión es exitoso, resuelve la promesa; en caso de error, rechaza la promesa con el mensaje de error.

Paso 3: Implementar el Método de Login en el Componente

Ahora que ya tenemos el servicio de login configurado, lo utilizamos en el componente de login. También se agregamos una variable para manejar los mensajes de error en caso de que el proceso de autenticación falle.

Código del archivo `login.component.ts`:

```

import { Component } from '@angular/core';
import { FormsModule } from '@angular/forms';
import { Router } from '@angular/router';
import { LoginService } from '../../servicios/login.service';

@Component({
  selector: 'app-login',
  standalone: true,
  imports: [FormsModule],
  templateUrl: './login.component.html',
}

```

```

    styleUrls: ['./login.component.css']
})
export class LoginComponent {
  email: string | null = null;
  password: string | null = null;
  mensaje: string | null = null;

  constructor(
    private router: Router,
    private loginService: LoginService
  ) { }

  login() {
    // Verificación de que email y password no son null
    if (this.email && this.password) {
      this.loginService.login(this.email, this.password)
        .then(() => {
          this.router.navigate(['/']);
        })
        .catch(error => {
          this.mensaje = 'Error al hacer login: ' + error;
        });
    } else {
      this.mensaje = 'Por favor, ingrese un email y una contraseña válidos.';
    }
  }
}

```

Explicación:

- El componente LoginComponent utiliza el servicio de login que creamos. Si el login es exitoso, el usuario es redirigido a la página de inicio.
- Si ocurre un error durante el login, se asigna un mensaje de error a la variable mensaje, que se mostrará en la interfaz.
- Se verifica que tanto el email como el password no sean nulos antes de intentar iniciar sesión.

Paso 4: Mostrar Mensajes de Error en la Interfaz

Para mejorar la experiencia del usuario, añadimos un mensaje de error estilizado con **Bootstrap** que aparece cuando las credenciales son incorrectas o faltan.

Código del archivo login.component.html:

```
<header id="main-header" class="py-2 bg-info text-white">
```

```
<div class="container">
  <div class="row">
    <div class="col-md-6">
      <h1><i class="bi bi-person"></i> Login</h1>
    </div>
  </div>
</header>

<!-- Mensaje de error centrado y estilizado con Bootstrap -->
<div class="container my-3">
  <div class="row">
    <div class="col-md-6 mx-auto">
      @if (mensaje) {
        <div class="alert alert-danger text-center" role="alert">
          {{ mensaje }}
        </div>
      }
    </div>
  </div>
</div>

<section id="login">
  <div class="container my-5">
    <div class="row">
      <div class="col-md-6 mx-auto">
        <div class="card">
          <div class="card-header">
            <h4>Login</h4>
          </div>
          <div class="card-body">
            <form (submit)="login()">
              <div class="mb-3">
                <label for="email" class="form-label">Email</label>
                <input
                  type="email"
                  class="form-control"
                  name="email"
                  [(ngModel)]="email"
                  required
                />
              </div>
              <div class="mb-3">
                <label for="password" class="form-label">Password</label>
                <input

```

```
        type="password"
        class="form-control"
        name="password"
        [(ngModel)]="password"
        required
      />
    </div>
    <input
      type="submit"
      value="Login"
      class="btn btn-primary w-100"
    />
  </form>
</div>
</div>
</div>
</div>
</div>
</section>
```

Explicación:

- Se ha añadido un bloque condicional `@if (mensaje)` para mostrar el mensaje de error si existe.
 - El mensaje de error se presenta dentro de un `div` con clases de **Bootstrap** para darle estilo y centrarlo en la pantalla. El `alert-danger` lo marca visualmente como un error.

Nota Final

Recordar que ya hemos configurado previamente el archivo main.ts, este archivo tiene la configuración del proveedor para poder autenticarnos al servicio de Cloud Firebase, así que por ello ya no es necesario modificar este archivo, sin embargo es importante recordar que allí se encuentra la configuración del proveedor de autenticación:

```
provideAuth(() => getAuth()), // Inicializa Auth
```

Conclusión

Con estos pasos, hemos implementado exitosamente el servicio de login en Angular usando Firebase y hemos manejado los mensajes de error de manera visualmente atractiva.

The screenshot shows the login interface of the 'Control Clientes' application. At the top, there's a dark header bar with the text 'Control Clientes' and 'Inicio' on the left, and 'Bienvenido' and 'Login' on the right. Below this is a bright blue header bar with the text '>Login'. The main content area has a dark background and contains a 'Login' form. The form includes fields for 'Email' (containing 'ubaldoacosta@gmail.com') and 'Password' (containing '.....'). A blue 'Login' button is at the bottom of the form. At the very bottom of the page is a yellow footer bar with the text '© GlobalMentoring.com.mx'.

Una vez autenticados correctamente, el usuario será redirigido a la pantalla principal donde podrá interactuar con el listado de clientes.

The screenshot shows the main dashboard of the 'Control Clientes' application. At the top, it features a dark header bar with 'Control Clientes' and 'Inicio' on the left, and 'Bienvenido' and 'Login' on the right. Below this is a bright blue header bar with a gear icon and the text 'Control de Clientes'. The main content area has a dark background. On the left, there's a table titled 'Listado de Clientes' showing three rows of client data: 'Carlos Lara' with a balance of '\$300.00', 'Juan Perez' with '\$100.00', and 'Karla Gomez' with '\$200.00'. Each row has an 'Editar' button. To the right of this table is a red box containing the text 'Saldo Total \$600.00'. Below this is a green box containing the text 'Total Clientes 3'. At the bottom of the page is a yellow footer bar with the text '© GlobalMentoring.com.mx'.

Recuerden que pueden descargar el código actualizado en la sección de recursos de esta lección.

¡Continúa así  , el esfuerzo vale la pena!

Saludos 

Ing. Ubaldo Acosta

Fundador de [GlobalMentoring.com.mx](http://www.globalmentoring.com.mx)