Taller2: Autenticación JWT

Autor

Daniela Erazo Marin

Docente

David Steven Duran Vallejo

Unidad Central Del Valle

Facultad De Ingeniería

Ingeniería De Sistemas

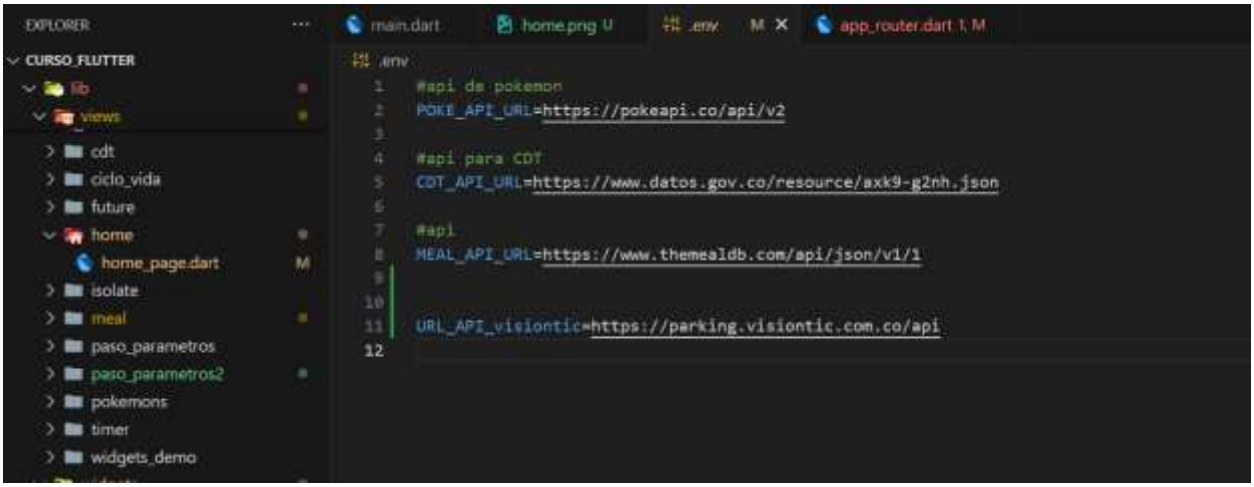
Tuluá – Valle Del Cauca

2025

Repo: https://github.com/daniela-erazo-marin/curso_flutter

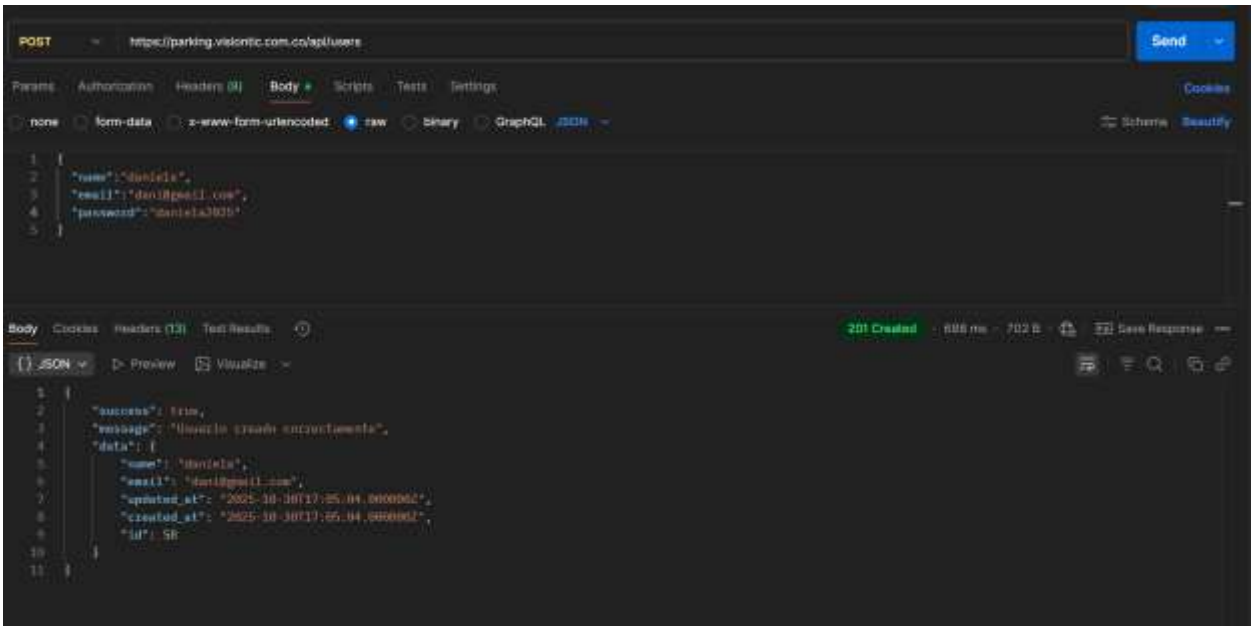
Autenticación JWT

Se implementa la autenticación contra la API pública https://parking.visiontic.com.co/api,

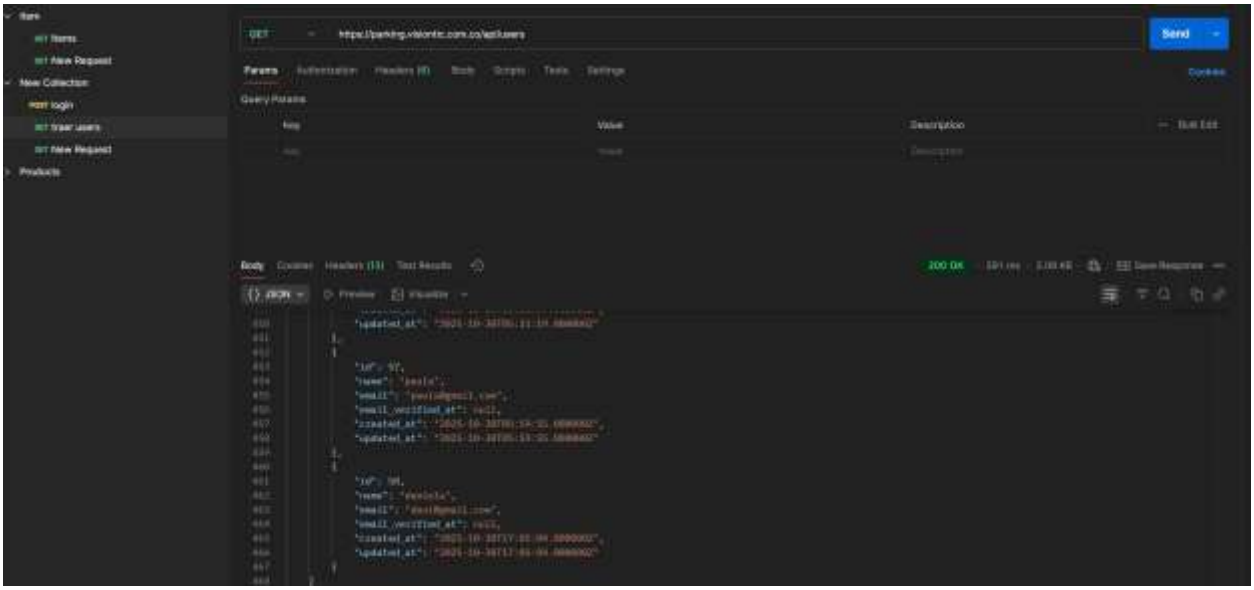


- Se creo un usuario por medio de Postman
- Se listaron los usuarios
- Se hizo un login de prueba en Postman
- Se obtuvo el perfil

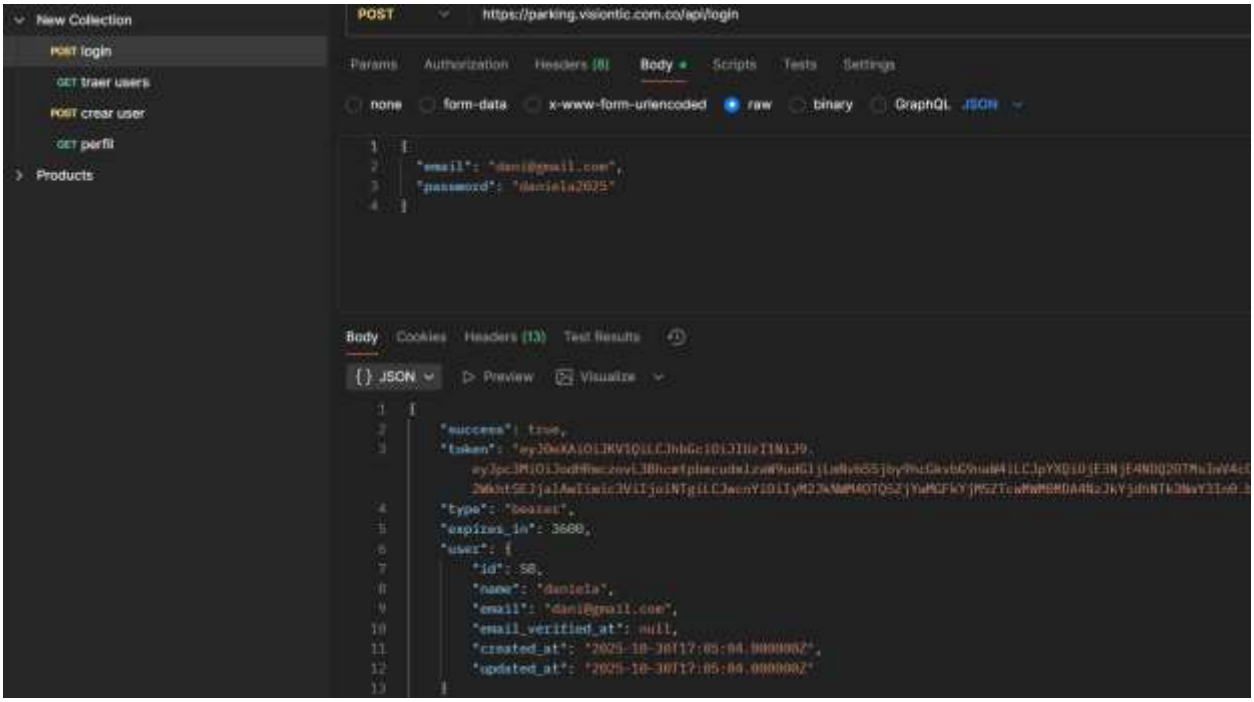
Crear user



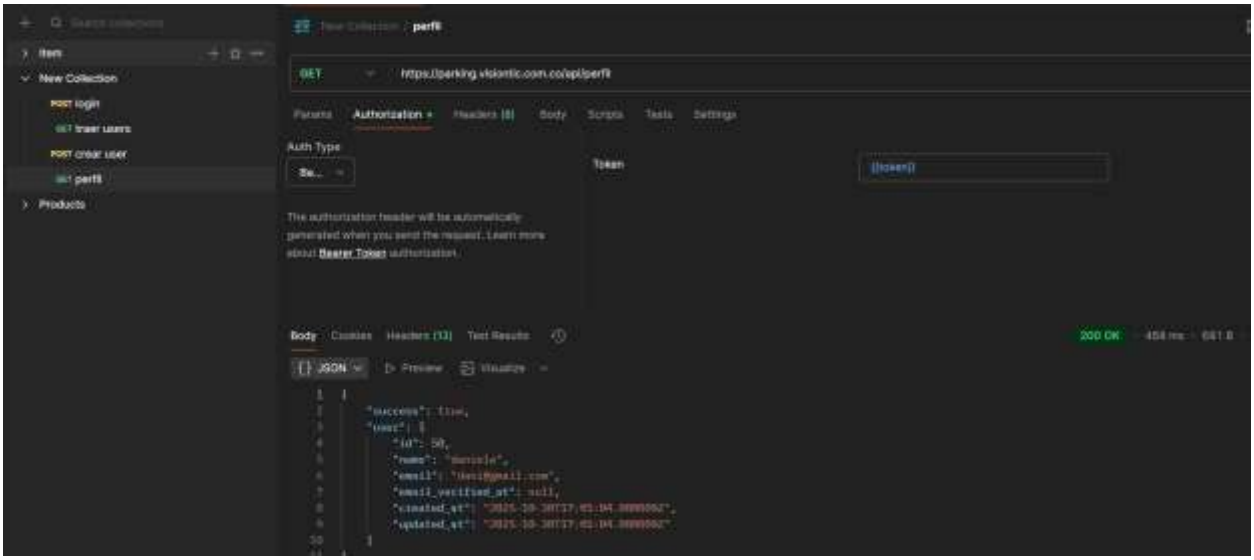
Lista de users



Login



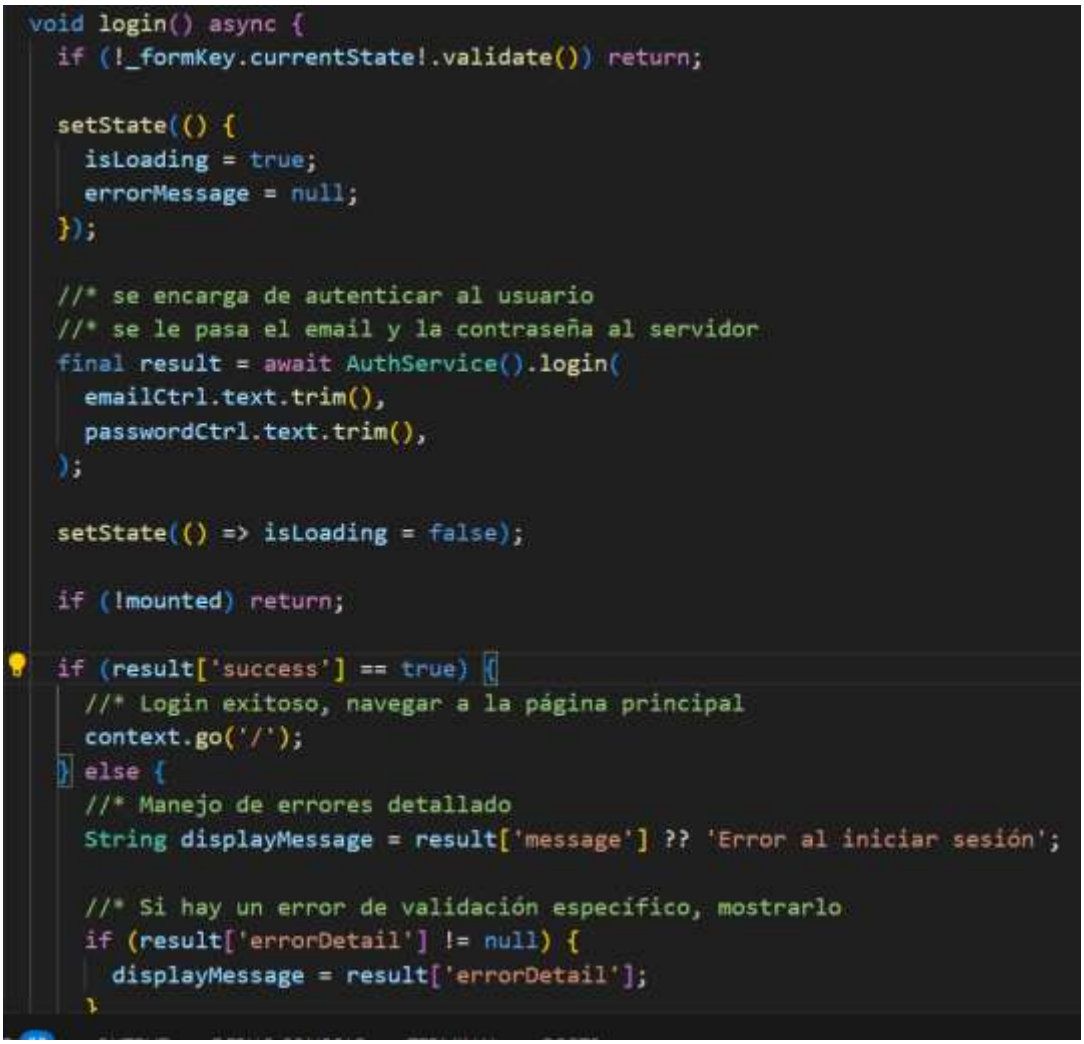
Obtener perfil



Se realiza el inicio de sesión

El método login() de la clase AuthService realiza la autenticación del usuario contra la API pública.

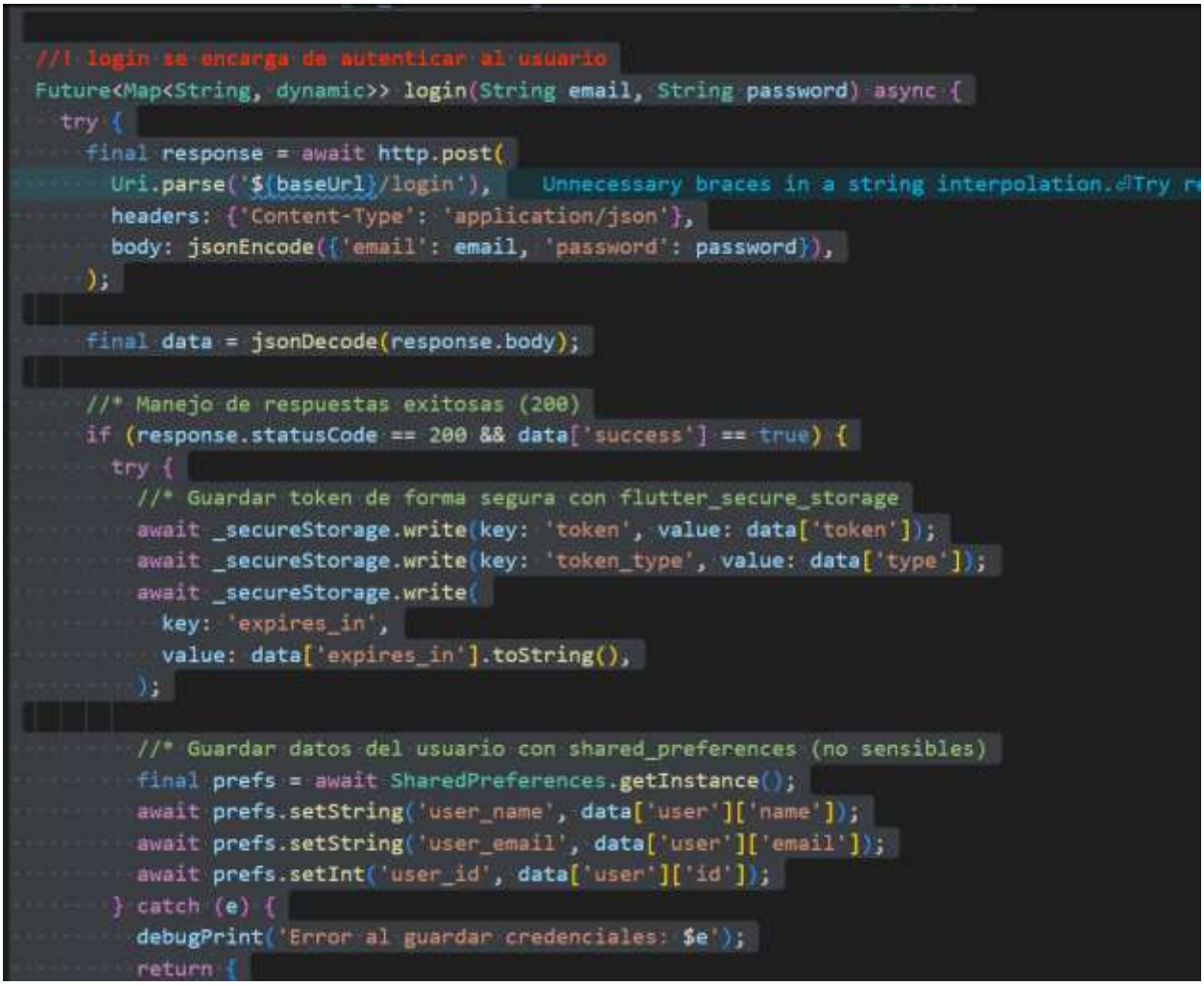
- Envía las credenciales (correo y contraseña) al endpoint /login.
- Si la autenticación es exitosa (statusCode 200 y success: true):
    - Guarda los datos sensibles (token, token_type, expires_in) en FlutterSecureStorage, de forma encriptada.
    - Guarda los datos no sensibles (user_name, user_email, user_id) en SharedPreferences, sin encriptar.



```
void login() async {
  if (!_formKey.currentState!.validate()) return;

  setState(() {
    isLoading = true;
    errorMessage = null;
  });

  //* se encarga de autenticar al usuario
  //* se le pasa el email y la contraseña al servidor
  final result = await AuthService().login(
    emailCtrl.text.trim(),
    passwordCtrl.text.trim(),
  );

  setState(() => isLoading = false);

  if (!mounted) return;

  if (result['success'] == true) {
    //* Login exitoso, navegar a la página principal
    context.go('/');
  } else {
    //* Manejo de errores detallado
    String displayMessage = result['message'] ?? 'Error al iniciar sesión';

    //* Si hay un error de validación especifico, mostrarlo
    if (result['errorDetail'] != null) {
      displayMessage = result['errorDetail'];
    }
```

- Si ocurre un error, se devuelve un mensaje detallado para mostrarlo en la interfaz de usuario.

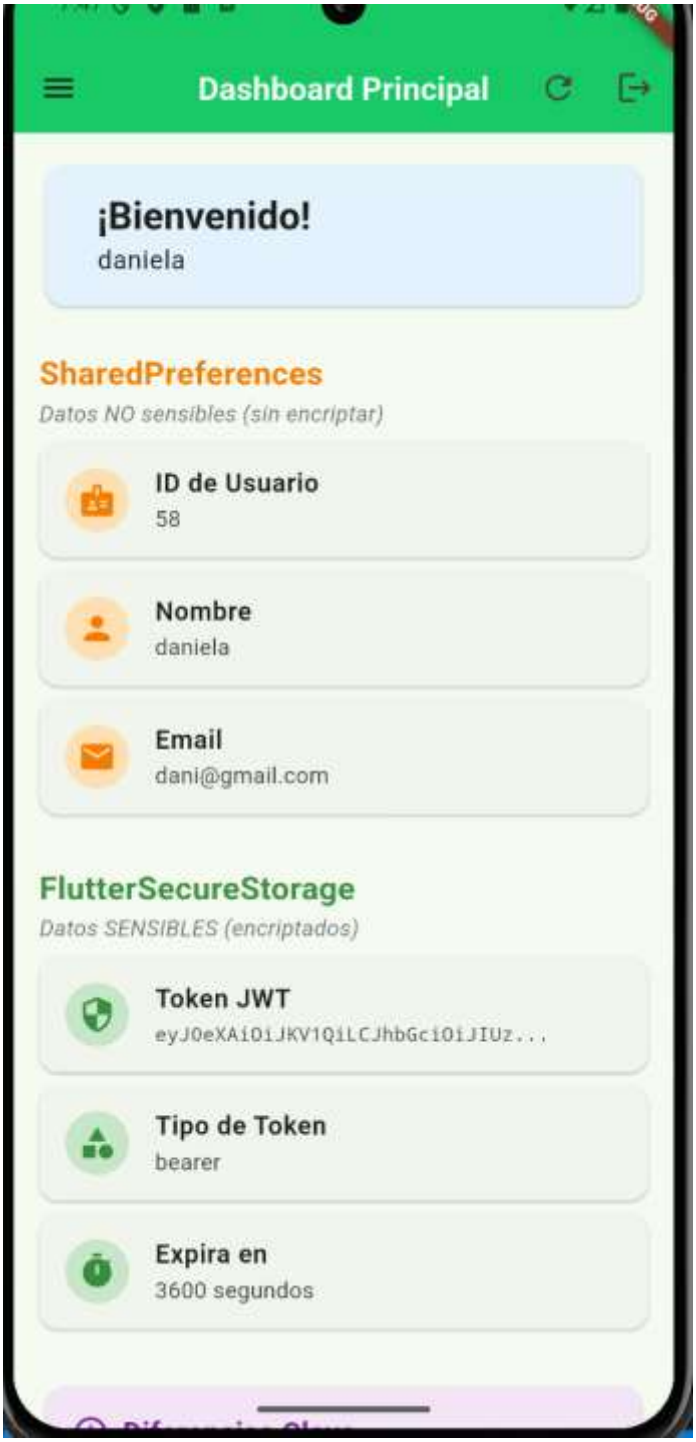
Login (API pública VisionTic)
Endpoint consumido: /api/login
Autenticación correcta → guarda token, token_type y expires_in en flutter_secure_storage (encriptado).
Guarda user_name, user_email, user_id en shared_preferences (texto plano).



```dart
//! login se encarga de autenticar al usuario
Future<Map<String, dynamic>> login(String email, String password) async {
  try {
    final response = await http.post(
      Uri.parse('${baseUrl}/login'),    Unnecessary braces in a string interpolation.dTry r
      headers: {'Content-Type': 'application/json'},
      body: jsonEncode({'email': email, 'password': password}),
    );

    final data = jsonDecode(response.body);

    //* Manejo de respuestas exitosas (200)
    if (response.statusCode == 200 && data['success'] == true) {
      try {
        //* Guardar token de forma segura con flutter_secure_storage
        await _secureStorage.write(key: 'token', value: data['token']);
        await _secureStorage.write(key: 'token_type', value: data['type']);
        await _secureStorage.write(
          key: 'expires_in',
          value: data['expires_in'].toString(),
        );

        //* Guardar datos del usuario con shared_preferences (no sensibles)
        final prefs = await SharedPreferences.getInstance();
        await prefs.setString('user_name', data['user']['name']);
        await prefs.setString('user_email', data['user']['email']);
        await prefs.setInt('user_id', data['user']['id']);
      } catch (e) {
        debugPrint('Error al guardar credenciales: $e');
        return {
```

Almacenamiento local requerido

- **shared_preferences** (NO sensible): nombre, email, tema/idioma, etc.
- **flutter_secure_storage** (Sensible): access_token



Cierre de sesión

El método logout() dentro de AuthService se encarga de eliminar toda la información almacenada del usuario.

- Borra los **datos sensibles** de FlutterSecureStorage:
    - token, token_type, expires_in.
- Borra los **datos no sensibles** de SharedPreferences:
    - user_name, user_email, user_id.
- Luego redirige al usuario a la pantalla de login (/login).