□ useAuth Hook - Usage Examples

Basic Import

```
import { useAuth } from '../hooks/useAuth';
```

Example 1: Basic Login Component

```
import { useState } from 'react';
import { useAuth } from '../hooks/useAuth';

function LoginForm() {
  const { login, isLoading, error } = useAuth();
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');

  const handleSubmit = async (e) => {
    e.preventDefault();
    const result = await login(email, password);

    if (result.success) {
      console.log('Login successful!', result.user);
      // Navigation is handled automatically by AuthService
    }
  };

  return (
    <form onSubmit={handleSubmit}>
      <input
        type="email"
        value={email}
        onChange={(e) => setEmail(e.target.value)}
        disabled={isLoading}
      />
      <input
        type="password"
        value={password}
        onChange={(e) => setPassword(e.target.value)}
        disabled={isLoading}
      />
      {error && <p className="error">{error}</p>}
      <button type="submit" disabled={isLoading}>
        {isLoading ? 'Logging in...' : 'Login'}
      </button>
    </form>
  );
}
```

Example 2: Registration Component

```
import { useState } from 'react';
import { useAuth } from '../hooks/useAuth';

function RegisterForm() {
  const { register, isLoading, error } = useAuth();
  const [formData, setFormData] = useState({
    nombre: '',
    email: '',
    telefono: '',
    password: '',
  });

  const handleSubmit = async (e) => {
    e.preventDefault();
    const result = await register(formData);

    if (result.success) {
      console.log('Registration successful!', result.user);
      // Auto-redirects to onboarding
    }
  };

  return (
    <form onSubmit={handleSubmit}>
      <input
        type="text"
        placeholder="Nombre"
        value={formData.nombre}
        onChange={(e) => setFormData({...formData, nombre: e.target.value})}
      />
      <input
        type="email"
        placeholder="Email"
        value={formData.email}
        onChange={(e) => setFormData({...formData, email: e.target.value})}
      />
      <input
        type="tel"
        placeholder="Teléfono"
        value={formData.telefono}
        onChange={(e) => setFormData({...formData, telefono: e.target.value})}
      />
      <input
        type="password"
        placeholder="Contraseña"
        value={formData.password}
        onChange={(e) => setFormData({...formData, password: e.target.value})}
      />
      {error && <p className="error">{error}</p>}
      <button type="submit" disabled={isLoading}>
        {isLoading ? 'Registering...' : 'Register'}
      </button>
    </form>
  );
}
```

Example 3: Protected Component with Role Check

```
import { useAuth } from '../hooks/useAuth';
import { Navigate } from 'react-router-dom';

function AdminPanel() {
  const { isAuthenticated, isAdmin, user } = useAuth();

  if (!isAuthenticated) {
    return <Navigate to="/login" />;
  }

  if (!isAdmin) {
    return <Navigate to="/dashboard" />;
  }

  return (
    <div>
      <h1>Admin Panel</h1>
      <p>Welcome, {user.nombre}!</p>
    </div>
  );
}
```

Example 4: User Profile Display

```
import { useAuth } from '../hooks/useAuth';

function UserProfile() {
  const { user, isAuthenticated } = useAuth();

  if (!isAuthenticated) {
    return <p>Please login to view profile</p>;
  }

  return (
    <div className="profile">
      <h2>{user.nombre}</h2>
```

```
        <p>Email: {user.email}</p>
        <p>Teléfono: {user.telefono}</p>
        <p>Rol: {user.rol}</p>
      </div>
    );
  }
```

Example 5: Conditional Rendering Based on Auth

```
import { useAuth } from '../hooks/useAuth';

function Navigation() {
  const { isAuthenticated, user, logout } = useAuth();

  return (
    <nav>
      {isAuthenticated ? (
        <>
          <span>Hello, {user.nombre}!</span>
          <button onClick={logout}>Logout</button>
        </>
      ) : (
        <>
          <a href="/login">Login</a>
          <a href="/register">Register</a>
        </>
      )}
    </nav>
  );
}
```

Example 6: Update Profile

```
import { useState } from 'react';
import { useAuth } from '../hooks/useAuth';

function EditProfile() {
  const { user, updateProfile, isLoading } = useAuth();
  const [formData, setFormData] = useState({
    nombre: user.nombre,
    telefono: user.telefono,
  });

  const handleSubmit = async (e) => {
    e.preventDefault();
    const result = await updateProfile(user.id, formData);
    if (result.success) {
      alert('Profile updated successfully!');
    }
  };

  return (
    <form onSubmit={handleSubmit}>
      <input
        type="text"
        value={formData.nombre}
        onChange={(e) => setFormData({...formData, nombre: e.target.value})}
      />
      <input
        type="tel"
        value={formData.telefono}
        onChange={(e) => setFormData({...formData, telefono: e.target.value})}
      />
      <button type="submit" disabled={isLoading}>
        Update Profile
      </button>
    </form>
  );
}
```

Example 7: Role-Based Menu

```
import { useAuth } from '../hooks/useAuth';
import { Link } from 'react-router-dom';

function SidebarMenu() {
  const { isAdmin, isConsumer } = useAuth();

  return (
    <aside>
      <ul>
        <li><Link to="/dashboard">Dashboard</Link></li>
        <li><Link to="/profile">Profile</Link></li>

        {isConsumer && (
          <>
            <li><Link to="/my-habits">My Habits</Link></li>
            <li><Link to="/progress">Progress</Link></li>
          </>
        )}

        {isAdmin && (
          <>
            <li><Link to="/admin/users">Manage Users</Link></li>
            <li><Link to="/admin/reports">Reports</Link></li>
          </>
        )}
      </ul>
    </aside>
  );
}
```

Example 8: Loading State

```
import { useAuth } from '../hooks/useAuth';

function Dashboard() {
  const { user, isLoading, isAuthenticated } = useAuth();

  if (isLoading) {
    return <div>Loading...</div>;
  }

  if (!isAuthenticated) {
    return <div>Please login</div>;
  }

  return (
    <div>
      <h1>Welcome, {user.nombre}!</h1>
    </div>
  );
}
```

Example 9: Refresh Token

```
import { useEffect } from 'react';
import { useAuth } from '../hooks/useAuth';

function App() {
  const { refreshToken, user } = useAuth();

  useEffect(() => {
    // Auto-refresh token every 14 minutes (if expires in 15min)
    const interval = setInterval(() => {
      if (user) {
        refreshToken();
      }
    }, 14 * 60 * 1000);

    return () => clearInterval(interval);
  }, [user, refreshToken]);
  return <div>Your app content</div>;
}
```

Example 10: Complete Auth Flow

```
import { useAuth } from '../hooks/useAuth';
import { useNavigate } from 'react-router-dom';

function CompleteAuthExample() {
  const navigate = useNavigate();
  const {
    user,
    isAuthenticated,
    isAdmin,
    isConsumer,
    login,
    logout,
    register,
    isLoading,
    error
  } = useAuth();

  const handleLogin = async (email, password) => {
    const result = await login(email, password);
    if (result.success) {
      if (result.redirectTo) {
        navigate(result.redirectTo);
      }
    }
  };

  const handleLogout = async () => {
    await logout();
    navigate('/login');
  };

  return (
    <div>
      {isLoading && <p>Loading...</p>}
      {error && <p className="error">{error}</p>}

      {isAuthenticated ? (
        <div>
          <h1>Welcome, {user.nombre}!</h1>
          <p>Email: {user.email}</p>
          <p>Role: {user.rol}</p>

          {isAdmin && <p>You have admin privileges</p>}
          {isConsumer && <p>You are a consumer</p>}
          <button onClick={handleLogout}>Logout</button>
        </div>
      ) : (
        <button onClick={() => handleLogin('test@example.com', 'password')}>
          Login
        </button>
      )}
    </div>
  );
}
```

📚 Available Properties & Methods

Properties

- `user` - Current user object (or null)
- `isAuthenticated` - Boolean
- `isAdmin` - Boolean (true if role === 'administrador')
- `isConsumer` - Boolean (true if role === 'consumidor')
- `isLoading` - Boolean
- `error` - String (error message or null)

Methods

- `login(email, password)` - Returns `{ success, user, redirectTo, error }`
- `register(userData)` - Returns `{ success, user, error }`
- `logout()` - Logs out and clears storage
- `refreshToken()` - Refreshes JWT token
- `updateProfile(userId, profileData)` - Updates user profile
- `checkAuth()` - Manually checks authentication status

💡 Tips & Best Practices

1. Always Check Authentication

```
const { isAuthenticated } = useAuth();

if (!isAuthenticated) {
  return <Navigate to="/login" />;
}
```

2. Handle Loading States

```
const { isLoading, user } = useAuth();
```

```
if (isLoading) return <Spinner />;
if (!user) return <LoginPrompt />;
```

3. Display Errors

```
const { error } = useAuth();

{error && <Alert type="error">{error}</Alert>}
```

4. Use Role Checks

```
const { isAdmin } = useAuth();

{isAdmin && <AdminFeatures />}
```

5. Clean Up on Unmount

```
useEffect(() => {
  // Subscribe to auth changes
  return () => {
    // Cleanup if needed
  };
}, []);
```

🚀 Quick Reference

```
// Import
import { useAuth } from '../hooks/useAuth';

// Use in component
const { user, isAuthenticated, login, logout } = useAuth();

// Login
await login('email@example.com', 'password');

// Logout
logout();

// Check role
if (isAdmin) { /* admin only code */ }

// Get user data
console.log(user.nombre, user.email);
```

Need more examples? Check `src/services/AuthService.js` for the underlying implementation!