Authentication Methods

# Overview
Supabase Authentication provides user management and security built on top of JSON Web Tokens (JWT). It supports multiple sign-in methods such as email/password, OAuth providers, and phone-based logins.

# Email/Password Sign-in
Example:
```
user = supabase.auth.sign_in(email="user@example.com", password="password123")
print(user)
```

# Sign-up
```
new_user = supabase.auth.sign_up(email="new@example.com", password="mypassword")
```

# Session Management
You can access the current session:
```
session = supabase.auth.get_session()
print(session.user.email)
```

# OAuth Providers
Supabase supports Google, GitHub, Facebook, and others:
```
supabase.auth.sign_in_with_provider("github")
```

# Magic Links
For passwordless login:
```
supabase.auth.sign_in(email="user@example.com")
```

# Reset Passwords
```
supabase.auth.api.reset_password_for_email("user@example.com")
```

# Notes on Security
Supabase uses JWT tokens to manage user sessions. These tokens can be verified on the backend for added security.

# Example Workflow
1. User signs up using email.
2. Supabase sends confirmation mail.
3. On successful confirmation, the user is logged in.
4. The app stores the JWT for subsequent API requests.

# Common Issues
- Expired tokens can cause “401 Unauthorized” errors.
- OAuth callback URLs must match your project settings.

# Summary
Supabase simplifies authentication with a consistent API across providers, ideal for integrating secure user management into Python applications.