OAuth2 Authorization Code Flow with PKCE

Participants: Client | User | Browser | Auth Server /oauth/authorize | Token Endpoint /oauth/token | Database

Step 1-2: Authorization Request → Consent

Client: Generate code_verifier, code_challenge (S256)

1. Redirect to /oauth/authorize

Browser → Auth Server: GET + PKCE challenge + state

Auth Server → Database: Validate client_id

Database → Auth Server: Client data

> ✓ PKCE S256 (oauth2_service.py:98)
> ✓ State validation (oauth_routes.py:46-70)

2. Show consent screen

Step 3: Authorization Code Response

3. User approves

Auth Server → Database: Store authorization code

302 Redirect + code + state

Callback with code

Step 4: Token Request with PKCE

4. POST /oauth/token + code + code_verifier

Token Endpoint → Database: BEGIN TRANSACTION

SELECT code WHERE used=0

Database → Token Endpoint: Code data

> ✓ PKCE: SHA256(verifier)==challenge
> ✓ Code unused (oauth2_service.py:180)
> ✓ Single-use enforcement (line 197)

UPDATE code SET used=1

Step 5: Token Response

INSERT tokens + token_family_id COMMIT

5. 200 OK: access_token, refresh_token (exp: 1h/30d)

Step 6: Resource Access

6. GET /oauth/userinfo Authorization: Bearer {token}

Validate access_token
Token valid, user data

200 OK: {sub, email, username}

> ✓ Token not expired (oauth2_service.py:282-285)
> ✓ Token not revoked (line 274)

Step 7: Token Refresh with Rotation

7. POST /oauth/token grant_type=refresh_token

BEGIN TRANSACTION
SELECT refresh_token
Check refresh_token_used

> ⚠ REUSE DETECTION (oauth2_service.py:320)
> If refresh_token_used=1 → REVOKE ENTIRE FAMILY
> Else: Mark old token used, issue new tokens with SAME family_id

UPDATE refresh_token_used=1
INSERT new tokens (SAME token_family_id) COMMIT

200 OK: new access_token, new refresh_token

Step 8: Token Revocation

8. POST /oauth/revoke token={access_or_refresh}

UPDATE SET revoked=1
Token revoked

200 OK (RFC 7009)

> Always return 200 even if token doesn't exist

Security Implementation Summary

PKCE Implementation:
- S256 method mandatory (oauth2_service.py:98)
- code_challenge validated during token exchange (oauth_routes.py:156-164)

Token Lifecycle:
- Authorization code: 10 minutes, single-use (line 23, 197)
- Access token: 1 hour, revocable (line 21)
- Refresh token: 30 days, rotation with family tracking (line 22, 227)

⚠ Critical: Refresh token reuse detection revokes entire token family (oauth2_service.py:320-325)