Vendor OAuth Integration Guide

This guide provides a detailed overview of the domains and API you need to successfully gain an access token to interact with the Total Expert API.

Required Configuration/Components

To successfully integrate with Total Expert (TE), the following items are needed:

- Your TE-created client ID and secret pair for OAuth 2.0 authentication
- Your TE-registered redirect URI
- Your development and/or production domains (hosting your redirect URI) registered with TE
- Your code that manages access token acquisition and use

The following sections describe these items and why they are essential.

Using the Total Expert Public API to Authenticate

TE will provide you with a client ID and secret pair unique to your integration for OAuth 2.0 authentication. Use your client ID and secret to obtain an access token for your user from the *public* domain. The authentication process follows these steps:

1. Your interface requests an authentication code with appropriate scope from the *public* domain's */authorize* endpoint (that is, *https://totalexpert.net/authorize*).
2. TE prompts your user with a Total Expert login page.
3. Log in to the Total Expert system with your TE credentials.
4. TE redirects to your preregistered callback URI and supplies the authentication code as a URL query parameter.
5. Your interface uses the authentication code to request an access token from the *public* domain's */v1/token* endpoint (that is, *https://public.totalexpert.net/v1/token*).

The "preregistered callback URI" (redirect URI) in step 4 is your custom callback. This callback must be supplied to TE to be associated with your client ID and secret. **You must register your redirect URI with TE for authentication to succeed.**

See the Public API: OAuth 2.0 Endpoints section below for the API request and response details and redirect URI query parameter examples.

Public API: OAuth 2.0 Endpoints

This section documents the public domain's API. Note that the production domain (*totalexpert.net*) is used in the endpoint examples.

GET Authentication Code

Obtain an OAuth 2.0 authentication code.

```
https://totalexpert.net/authorize?response_type=code&client_id=AppClientID&scop
e=crm&state=myState
```

Query Parameters

This endpoint supports the following query parameters:

Parameter	Required?	Description
`response_type`	yes	Always "code".
`client_id`	yes	The application's ID assigned by Total Expert.
`scope`	yes	A space-delimited list of strings (for example "leadSurveyInteraction postLeads") that defines access that the application is requesting.
`state`	yes	The application's state. This is not modified and is returned to the redirect callback with the code.
`redirect_uri`	no	The application's redirect callback URI. This must match the registered string, which is the value used if this parameter is not supplied.
`serviceProvider`	no	The client's SSO service provider to be used for single sign on services.

Expected Behavior/Responses

If the request is valid, the API returns a login page. Once the login is processed, the API redirects the caller to the registered `redirect_uri`, appending a certain set of query parameters to the URI.

The following is an example of a redirect URI for a successful login:

```
https://your.domain.com/oauth2/callback?code=bc9fc71b592e7e2bbbcc9453a2f0540b26
8ce7f5&state=myState
```

If the request is invalid, or if the user denies or fails the login attempt, the redirect still occurs, but the URI has different parameters. The following is an example of a redirect URI for a failed authentication request:

```
https://your.domain.com/oauth2/callback?error=access_denied&error_description=T
he+user+denied+access+to+your+application&state=myState
```

The query parameters that can appear in callback URIs are:

Parameter	When Included	Description
`code`	On success	Authorization code
`state`	Always	Application's state
`error`	On failure	Reason for failure
`error_description`	On failure	Description of failure

POST Fetch an Access Token

Obtain an OAuth 2.0 access token.

`https://public.totalexpert.net/v1/token`

Using an Authentication Code

This request requires the *Authorization* header and a body. The *Authorization* header should contain the string `Basic {{basic_auth}}` where `{{basic_auth}}` is the base64-encoded string `appClientID:appClientSecret`. The body should contain two key/value pairs:

- grant_type: `authorization_code`
- code: `{{authorization code}}`, where `{{authorization code}}` is the value sent to the redirect URI in the "code" query parameter

Using a Refresh Token

An access token is only good for a certain amount of time, after which it expires and cannot be used to access the system. If the client needs a new access token, the client can use the `refresh_token` returned from */token* to request a new access token.

This request also requires the *Authorization* header and a body. The *Authorization* header should contain the string `Basic {{basic_auth}}` where `{{basic_auth}}` is the base64-encoded string `appClientID:appClientSecret`. The body should contain two key/value pairs:

- grant_type: `refresh_token`
- refresh_token: `{{refresh token}}`, where `{{refresh token}}` is the `refresh_token` value from the original response

Expected Behavior/Responses

If the token is granted, the response JSON body contains:

Parameter	Description
`access_token`	A token to include in the Authorization header when accessing the system.
`expires_in`	Time, in seconds, until the token expires. If attempting to access the system with an expired token, access will be denied.
`token_type`	Always "Bearer".
`scope`	A space-delimited list of strings that define the access scope of the token.
`refresh_token`	A token used to request a new access token.

If the request is invalid or fails, the response will include an `error` and an `error_description` in the response JSON body.

SSO

Total Expert supports SSO capabilities. For your integration to leverage those SSO capabilities, you must also pass Total Expert the service provider name:

```
https://totalexpert.net/authorize?serviceProvider=<providerName>
```

This service provider name should be configurable in your platform and passed when requesting the OAuth token.