ID.NOT Document d'intégration OpenIDConnect

Version **V1.0**

Table des matières

1 Glossaire

Terme	Description
ID.NOT (Identité numérique notariale)	Solution d'authentification centralisé et unique pour les applications de la profession notariale.
OpenID Connect (OIDC)	Protocole d'**authentification** basé sur OAuth 2.0 qui ajoute une **couche d'identité**. Permet de vérifier l'identité d'un utilisateur.
Identity Provider (IdP)	Serveur qui **authentifie** l'utilisateur et fournit des **tokens** aux applications (RPs). ID.NOT joue le rôle de l'IdP
Relying Party (RP)	Application cliente qui fait **confiance** à l'IdP pour gérer l'authentification de ses utilisateurs.
ID Token	**JWT** contenant l'identité de l'utilisateur. Sert à **authentifier** auprès du RP.
Access Token	Token qui permet d'accéder à une **API protégée**. Sert à **autoriser**, pas à authentifier.
Refresh Token	Token permettant de **renouveler un Access Token** sans ré-authentification de l'utilisateur.
Claims	Informations **sur l'utilisateur** contenues dans les tokens (ex : `sub`, `name`, `email`).
Subject (`sub`)	Identifiant **unique** d'un utilisateur attribué par l'IdP.
Authorization Endpoint	URL de l'IdP où l'utilisateur est redirigé pour **s'authentifier**.
Token Endpoint	URL de l'IdP où le RP échange le **code d'autorisation** contre des **tokens**.
Authorization Code Flow	**Flux standard** pour obtenir des tokens via **un code d'autorisation** (sécurisé côté serveur).
PKCE	Mécanisme renforçant la sécurité du **Authorization Code Flow**, surtout pour les applis publiques (SPA, mobiles).
Scopes	Permissions demandées par le RP à l'IdP (ex : `openid`, `profile`, `email`).
Client ID	Identifiant public d'une application enregistrée auprès de l'IdP.
Client Secret	Mot de passe partagé entre le RP et l'IdP (usage côté serveur uniquement).
Single Logout (SLO)	Déconnexion **globale** de toutes les sessions utilisateurs dans les RPs après logout SSO.
RP-Initiated Logout	Le RP demande à l'IdP de **terminer la session SSO** (Frontchannel Logout).
Frontchannel Logout	**Déconnexion via navigateur** : l'IdP notifie les RPs avec des **redirections** ou **iframes**.

Terme	Description
Backchannel Logout	Déconnexion **serveur à serveur** : l'IdP notifie directement les RPs via **HTTP POST**, sans intervention utilisateur.
Logout Token	JWT envoyé par l'IdP à chaque RP pour **notifier une déconnexion** pendant un Backchannel Logout.
Session ID (`sid`)	Identifiant de session partagé entre l'IdP et les RPs, utile pour les mécanismes de logout.
Discovery Document	URL `/.well-known/openid-configuration` qui fournit les **infos de configuration** de l'IdP.
JWKS (JSON Web Key Set)	Ensemble de **clés publiques** de l'IdP permettant de vérifier les signatures des JWT.

2 Configuration de l'Idp ID.NOT

Le **"Discovery Document"** est un document JSON que publie l'**Identity Provider (IdP)** pour **décrire automatiquement sa configuration**.
Son but est de permettre à un **client** (SPA, mobile, backend…) de **découvrir dynamiquement** :

- **Les endpoints** (URL de login, token, userinfo, etc.)
- **Les algorithmes de signature**
- **Les fonctionnalités supportées** (scopes, claims, etc.)

Discovery Document	https://connexion.idnot.fr/IdPOAuth2/idnot_idp_v1/.well-known/openid-configuration

3 Flux OpenIDConnect

ID.NOT supporte les deux flux **Authorization Code Flow** et **Authorization Code Flow avec PKCE** (Proof Key for Code Exchange) dans le parcours d'authentification de l'utilisateur.

Les deux flux font partie du protocole OAuth 2.0 et sont utilisés pour obtenir un **code d'autorisation** et l'échanger contre un **jeton d'accès**. Cependant, leur utilisation dépend du type d'application qui en fait la demande et du niveau de sécurité requis.

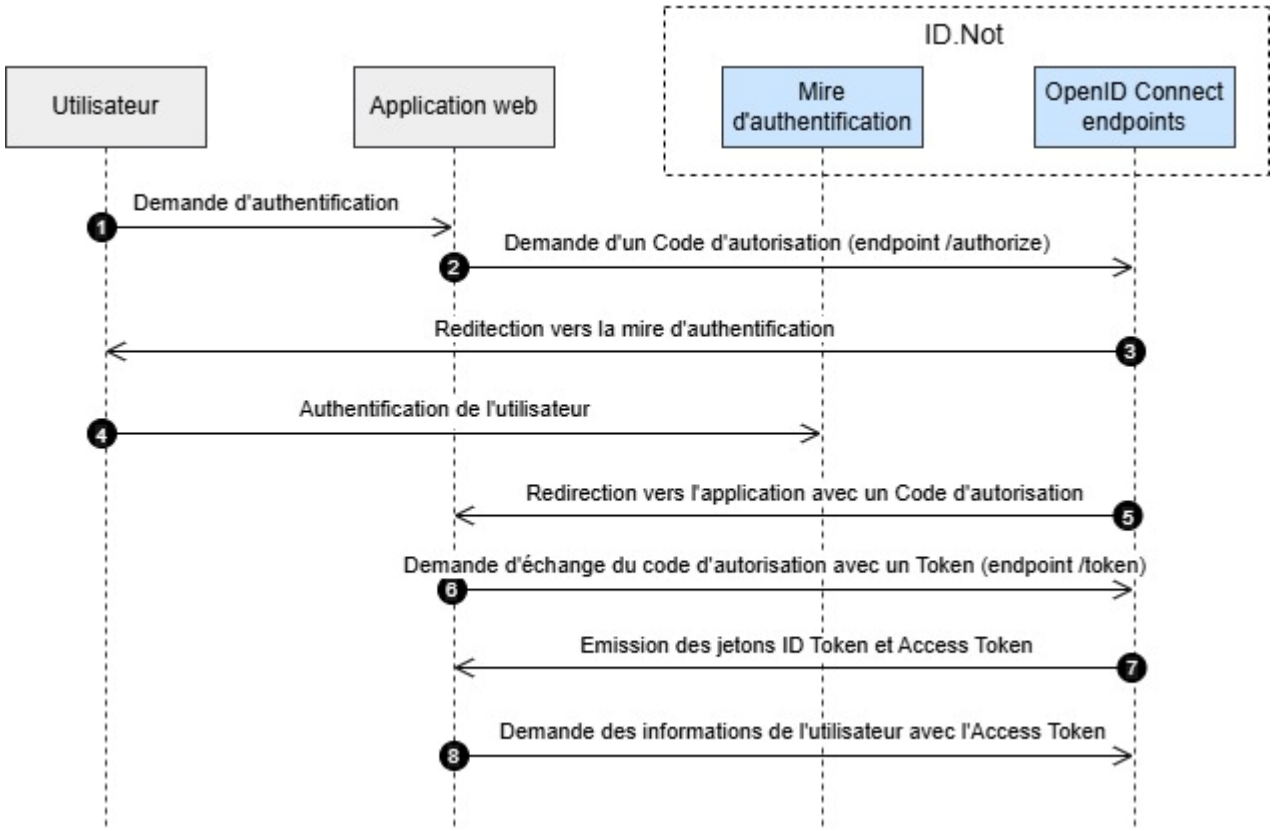
*Les flux **Implicit Flow** et **Hybrid Flow** sont **dépréciés** et **non recommandés** pour la plupart des cas d'usage.*

4 Login - Authorization Code Flow

Ce flux est généralement utilisé pour des **applications côté serveur** ou des applications **confidentielles** (celles qui peuvent garder un secret, comme une application web côté serveur).

Quand l'utiliser :

- Lorsque vous avez un contrôle total sur l'environnement backend et que vous pouvez protéger les informations sensibles (comme le **client_secret**).



Etape 1: Demande d'authentification et d'autorisation (Authorization Request)

L'utilisateur doit d'abord être redirigé vers ID.NOT pour effectuer l'authentification. Voici un exemple d'URL de requête :

```
GET https://connexion.idnot.fr/IdPOAuth2/authorize/idnot_idp_v1?
    response_type=code&
    client_id=your-client-id&
    redirect_uri=https://your-application.com/callback&
    scope=openid profile email&
    state=xyz123&
    nonce=abc456
```

Une fois l'utilisateur authentifié, ID.NOT redirige l'utilisateur vers l'URL de redirection spécifiée, avec un **code d'autorisation** temporaire**.** La durée de validité du code est 30 secondes.

Etape 2: Échange du code d'autorisation contre un jeton d'accès (Token Request)

*Dans le cas du flux **Authorization Code**, la **clé secrète** est utilisée pour **authentifier** l'application.*

```
POST https://connexion.idnot.fr/IdPOAuth2/token/idnot_idp_v1
Content-Type: application/x-www-form-urlencoded

grant_type=authorization_code&
code=authorization_code_received_from_redirect&
redirect_uri=https://your-application.com/callback&
client_id=your-client-id&
client_secret=your-client-secret
```

La réponse est sous format JSON

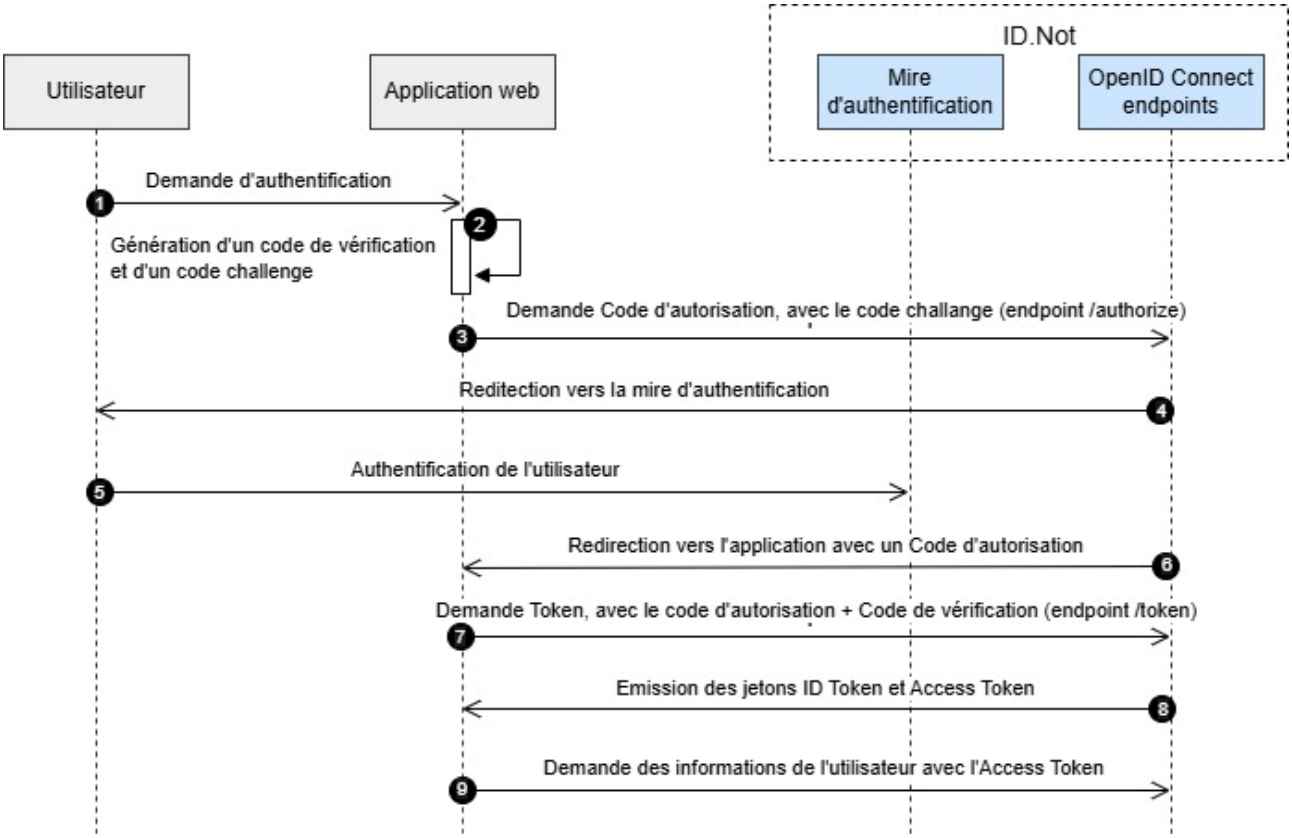
```
{
  "access_token": "access_token_value",
  "id_token": "id_token_value",
  "token_type": "bearer",
  "expires_in": 3600,
}
```

5 Login - Autorization Code Flow With PKCE

Le flux **Authorization Code Flow avec PKCE** est conçu pour améliorer la sécurité dans les cas où l'application cliente **ne peut pas** garder un **secret** en toute sécurité. Cela inclut les applications **côté client**, comme les applications **mobiles** ou les **applications JavaScript**.

Quand l'utiliser :

- Lorsque vous développez des applications **côté client** qui ne peuvent pas garder un secret (comme les **applications web SPA** ou **applications mobiles**).
- Pour **accroître la sécurité** de l'échange de jetons, notamment pour protéger contre les attaques de type "code interception".
- Améliorer la sécurité sans l'utilisation de la clé secrète.



Etape 1: Demande d'authentification et d'autorisation (Authorization Request)

L'utilisateur doit d'abord être redirigé vers ID.NOT pour effectuer l'authentification. Voici un exemple d'URL de requête :

```
GET https://connexion.idnot.fr/IdPOAuth2/authorize/idnot_idp_v1 ?
    response_type=code&
    client_id=your-client-id&
    redirect_uri=https://your-application.com/callback&
    scope=openid profile email&
    state=xyz123&
    nonce=abc456&
    code_challenge=code_challenge_value&
    code_challenge_method=S256
```

Une fois que l'utilisateur s'authentifie, ID.NOT redirige l'utilisateur vers l'URL de redirection spécifiée, avec un **code d'autorisation** temporaire**.** La durée de validité du code est 30 secondes.

Etape 2: Échange du code d'autorisation contre un jeton d'accès (Token Request)

*Dans le flux **Authorization Code avec PKCE,** le **code de vérification** sert à **protéger l'échange du code d'autorisation contre un jeton d'accès**, en particulier contre les attaques de type "code interception. PKCE est utilisé pour améliorer la sécurité sans secret.*

```
POST
https://connexion.idnot.fr/IdPOAuth2/token/idnot_idp_v1
Content-Type: application/x-www-form-urlencoded

grant_type=authorization_code&
code=authorization_code_received_from_redirect&
redirect_uri=https://your-application.com/callback&
client_id=your-client-id&
code_verifier=code_verifier_value
```

La réponse est sous format JSON

```
{
  "access_token": "access_token_value",
  "id_token": "id_token_value",
  "token_type": "bearer",
  "expires_in": 3600,
}
```

6 Vérification de la signature du Token

La vérification de la signature de l'**ID Token** permet de s'assurer que :

1. Le token vient **bien de l'IdP** (authenticité)
2. Il n'a pas été **modifié** (intégrité)

Vous trouverez la clé publique et les paramètres nécessaires à la vérification de la signature dans dans le **endpoint jwks_uri**

```
"jwks_uri":"https://connexion.idnot.fr/user/IdPOAuth2/jwk/idnot_idp_v1"
```

7 Données utilisateur

Les données de l'utilisateur sont portées par l'ID Token.
Le jeton d'identification contient les informations suivantes

- **Sub**: identifiant unique de l'utilisateur.
- **Name** : Nom usuel de l'utilisateur.
- **Given_name**: prénom de l'utilisateur.
- **Entity_idn**: identifiant de la structure liée au profil de connexion.
- **Profile_idn**: identifiant du rattachement lié au profil de connexion.
- Le rattachement est le lien entre la personne et son entité, il porte la fonction et les données professionnelles.
- **Email**: email professionnel (donnée du rattachement)
    - Information non présente par défaut. Le besoin doit être justifié.

Toute information complémentaire peut être récupérée via l'API Annuaire (https://api.notaires.fr/annuaire) en utilisant l'identifiant unique de l'utilisateur (**claim Sub**), l'identifiant du profil de connexion (**profile_idn**) ou l'identifiant de la structure (**entity_idn**)

Exemple

HEADER:ALGORITHM & TOKEN TYPE

```
{
  "alg": "HS512"
}
```

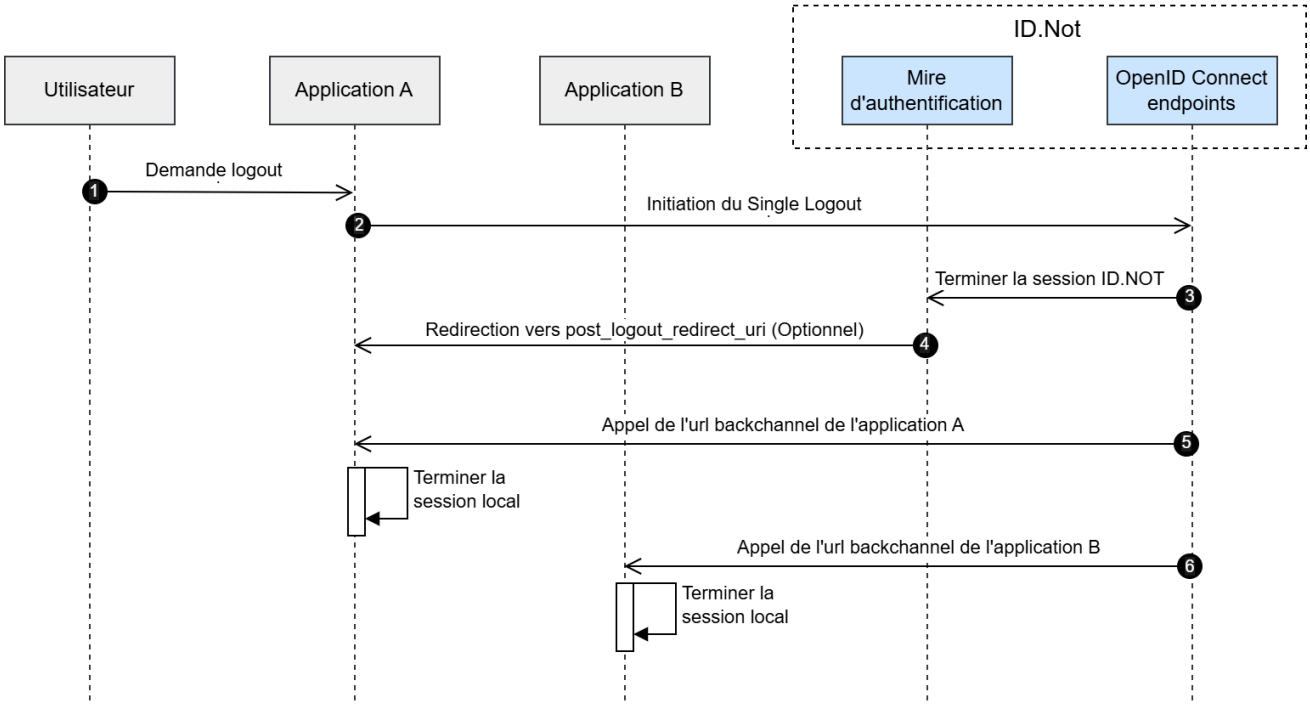
PAYLOAD:DATA

```
{
  "at_hash": "Xsv3DwIEh1BkVUR2kdYeDIhFrZQTk0-
6dp5sZzv-ma0",
  "sub": "IDN58541659I",
  "profile_idn": "IDN58541659I_IDN490058",
  "amr": [
    "3"
  ],
  "iss":
"https://connexion.idnot.fr/IdPOAuth2/idnot_idp_v1",
  "given_name": "Patrick",
  "sid": "m88yxfv89-320",
  "aud": "0CBF8BA6C28C0B54",
  "nbf": 1741968315,
  "auth_time": 1741968313,
  "entity_idn": "IDN490058",
  "name": "Martin",
  "exp": 1741975515,
  "iat": 1741968315,
  "email": "patric.martin@sous-domaine.notaires.fr"
}
```

VERIFY SIGNATURE

```
HMACSHA512(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
)
```

8 Logout - Déconnexion unique



8.1 Initiation du SLO par le RP (RP-Initiated Logout)

Quand on parle de **RP-Initiated Logout** dans **OpenID Connect**, c'est le **Relying Party (RP)** (l'application) qui **initie la déconnexion** de l'utilisateur auprès de l'**Identity Provider (IdP)** suivant les étapes ci-dessous :

1. L'utilisateur se déconnecte d'un **RP**.
2. Le **RP** demande à l'**IdP** de déclencher la déconnexion centralisée.
3. L'IdP peut ensuite **rediriger** l'utilisateur vers une URL de post-logout (facultatif).

Requête

```
GET https://connexion.idnot.fr/IdPOAuth2/logout/idnot_idp_v1
    ?id_token_hint=eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9...
    &post_logout_redirect_uri=https%3A%2F%2Frp.example.com%2Flogout-callback
```

Paramètres de la requête

Paramètre	Description
`id_token_hint`	L'ID Token précédemment délivré par l'IdP, pour identifier la session de l'utilisateur.
`post_logout_redirect_uri`	(Optionnel) L'URL vers laquelle l'IdP va rediriger l'utilisateur après la déconnexion. Doit être pré-enregistrée dans l'IdP.

8.2 Backchannel logout

ID.NOT supporte le mécanisme **Backchannel Logout** qui permet à un **Identity Provider (IdP)** de notifier silencieusement une **application** de la déconnexion de l'utilisateur.

Contrairement à **Frontchannel Logout**, l'utilisateur **n'est pas impliqué** dans la redirection ; c'est une **communication serveur-à-serveur** (backchannel).
Ce mécanisme garantit que :

- Les **sessions locales** sont fermées **en temps réel**.
- La **cohérence et la sécurité** des sessions sont respectées **sur toutes les applications** raccordées à ID.NOT.

Si votre application implémente le **endpoint backchannel logout** et que celui-ci est enregistré auprès de ID.NOT, votre application sera notifiée pour toute déconnexion

1. Déconnexion initiée par votre application
2. Déconnexion initiée par une autre application raccordée à ID.NOT.
3. Déconnexion initiée par ID.NOT.

9 Authentification silencieuse

L'**authentification silencieuse** permet de vérifier si un utilisateur est déjà connecté sur un **Identity Provider** (IdP), **sans interaction utilisateur** ni affichage de formulaire de login.

L'objectif est de récupérer **un token** (ou un nouveau token) **en arrière-plan**, sans perturber l'utilisateur, souvent pour maintenir une session active.

Dans **OpenID Connect**, lorsqu'une application fait une requête d'**autorisation** vers l'IdP, il est possible d'ajouter le paramètre :

```
prompt=none
```

Ce que signifie :

- **none** indique au serveur **de ne pas afficher d'interface utilisateur**.
- Si l'utilisateur **est déjà authentifié**, le serveur retourne directement un **code d'autorisation** ou un **token**.
- Si l'utilisateur **n'est pas authentifié**, ID.NOT retourne une **erreur login_required**.

*Le **flux Authorization Code avec PKCE** est **le plus utilisé** pour l'**authentification silencieuse** en OpenID Connect.*
*Il est plus sécurisé **et compatible SPA.***

*Les flux **Implicit Flow** et **Hybrid Flow** sont **dépréciés** et **non recommandés** pour la plupart des cas d'usage.*

Etape 1: Demande d'authentification et d'autorisation (Authorization Request)

GET https://connexion.idnot.fr/IdPOAuth2/authorize/idnot_idp_v1
?response_type=code
&client_id=abc
&redirect_uri=https://app.com/callback
&scope=openid profile
&**prompt=none**
&code_challenge=xyz
&code_challenge_method=S256

- Si l'utilisateur possède une session ID.NOT, il est redirigé vers "**redirect_uri**", avec un **code d'autorisation.**
- Si non, ID.NOT redirige l'utilisateur vers "**redirect_uri**" avec un message d'erreur

https://app.com/callback#error=login_required&error_description=login_required

Etape 2: Échange du code d'autorisation contre un jeton d'accès (Token Request)

```
POST
https://connexion.idnot.fr/IdPOAuth2/token/idnot_idp_v1
Content-Type: application/x-www-form-urlencoded
client_id=abc
grant_type=authorization_code
code=received_code
redirect_uri=https://app.com/callback
code_verifier=xyz_original
```