OpenID example configurations Vikunja

Link: https://vikunja.io/docs/openid-example-configurations/
EM 22/08/2026

On this page you will find examples about how to set up Vikunja with a third-party OAuth 2.0 provider using OpenID Connect. To add another example, please edit this document and send a PR.

Important: Redirect URL Format The redirect URL format is: `https://vikunja.mydomain.com/auth/openid/<provider-id>` Use the provider ID (the key in your config) as the identifier. For example, if your config uses `authentiklogin:` as the provider key, the redirect URL would be `/auth/openid/authentiklogin`.

Authelia#

Vikunja Config:

```
auth:
  openid:
    enabled: true
    providers:
      authelia:
        name: Authelia
        authurl: https://login.mydomain.com
        clientid: <vikunja-id>
        clientsecret: <vikunja secret>
```

Authelia config:

```
- client_id: <vikunja-id>
  client_name: Vikunja
  client_secret: <vikunja secret>
  redirect_uris:
```

```
    - https://vikunja.mydomain.com/auth/openid/authelia
  scopes:
    - openid
    - email
    - profile
```

Also see the Authelia documentation.

Google / Google Workspace#

Vikunja Config:

```
auth:
  openid:
    enabled: true
    providers:
      google:
        name: Google
        authurl: https://accounts.google.com
        clientid: <google-oauth-client-id>
        clientsecret: <google-oauth-client-secret>
```

Google config:

- Navigate to `https://console.cloud.google.com/apis/credentials` in the target project
- Create a new OAuth client ID
- Configure an authorized redirect URI of `https://vikunja.mydomain.com/auth/openid/google`

Note that there currently seems to be no way to stop creation of new users, even when `enableregistration` is `false` in the configuration. This means that this approach works well only with an “Internal Organization” app for Google Workspace, which limits the allowed users to organizational accounts only. External / public applications will potentially allow every Google user to register.

Keycloak#

Vikunja Config:

```
auth:
  openid:
    enabled: true
    providers:
      keycloak:
        name: Keycloak
        authurl: https://keycloak.mydomain.com/realms/<realm-name>
        logouturl: https://keycloak.mydomain.com/realms/<realm-name>/protocol/openid-
connect/logout
        clientid: <vikunja-id>
        clientsecret: <vikunja secret>
```

Keycloak Config:

- Navigate to the keycloak instance
- Create a new client with the type `OpenID Connect`, add a unique `Client ID`.
- Set `Client authentication` to On
- Set `Root Url` to `https://vikunja.mydomain.com`
- Set `Valid redirect URIs` to `/auth/openid/keycloak`
- Create the client then navigate to the credentials tab and copy the `Client secret`

Authentik#

Authentik Config:

- Create a new Provider called “Vikunja” in Authentik
- Set the `Redirect URIs/Origins (RegEx)` to `https://vikunja.mydomain.com/auth/openid/authentik` (This matches the `name: authentik` in the Vikunja config below)
- Copy the Client ID and Client Secret

Vikunja Config:

```
auth:
  openid:
    enabled: true
    providers:
      authentik:
        name: authentik
        authurl: "https://authentik.mydomain.com/application/o/vikunja/"
```

```
        logouturl: "https://authentik.mydomain.com/application/o/vikunja/end-session/"
        clientid: "" # copy from Authentik
        clientsecret: "" # copy from Authentik
```

Note: The `authurl` that Vikunja requires is not the `Authorize URL` that you can see in the Provider. OpenID Discovery is used to find the correct endpoint to use automatically, by accessing the `OpenID Configuration URL` (usually `https://authentik.mydomain.com/application/o/vikunja/.well-known/openid-configuration`). Use this URL without the `.well-known/openid-configuration` as the `authurl`. Typically, this URL can be found in the metadata section within your identity provider.

Azure Entra ID#

Vikunja Config:

```
auth:
  openid:
    enabled: true
    providers:
      azuread:
        name: AzureAD
        authurl: https://sts.windows.net/<tenant_id>/
        clientid: <azure-client-id>
        clientsecret: <azure-client-secret>
```

Azure AD Config:

- Navigate to the Azure Portal and create a new App Registration
- Set the Redirect URI to `https://vikunja.mydomain.com/auth/openid/azuread`
- Under API permissions, add the following delegated API permissions:
  - email
  - openid
  - profile
  - User.Read
- Create a new client secret and copy its value
- In Token configuration, add an optional claim:
  - Select ID, then email

Note: Replace `<tenant_id>` in the `authurl` with your Azure AD tenant ID. Ensure that the provider name in the Vikunja config matches the one used in the redirect URI (e.g., “azuread” in this example).

Dex#

Dex config:

```
staticClients:
- id: <vikunja-id>
  redirectURIs:
    - 'https://vikunja.mydomain.com/auth/openid/dex'
  name: 'Vikunja'
  secret: <dex-client-secret>
```

Note: All scopes required by Vikunja are already in the default scope definition of Dex (see Dex docs).

Vikunja config:

```
auth:
  openid:
    enabled: true
    providers:
      dex:
        name: dex
        authurl: <dex-url>
        clientid: <vikunja-id>
        clientsecret: <dex-client-secret>
        scope: openid profile email
        forceuserinfo: false
```

Revision #2
Created 2026-08-22 21:09:30 UTC by Administrador
Updated 2026-08-22 21:11:58 UTC by Administrador