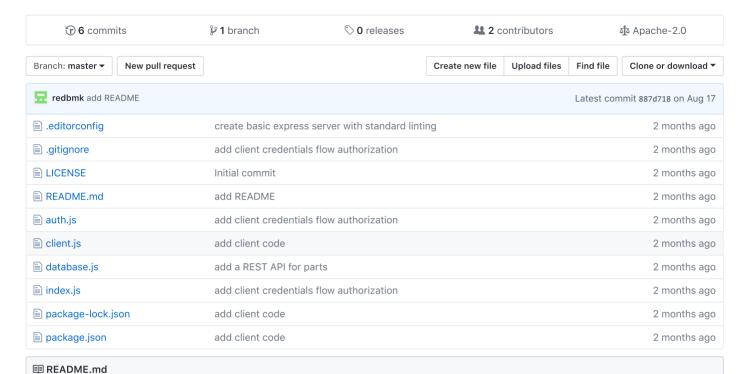
oktadeveloper / okta-node-rest-api-example

Simple REST API with Node and OAuth 2.0 https://developer.okta.com/blog/2018/...

#rest-api #rest #nodejs #oauth2 #javascript



Simple REST API with Node and OAuth 2.0

This example app shows how to create a REST API in Node and secure it with OAuth 2.0 Client Credentials using Okta. This also has an example client written as a CLI that can authenticate with Okta and use the REST API.

Getting Started

Install Dependencies

After cloning the repository, simply run npm install to install the dependencies.

Save Environment Variables

If you don't have one already, sign up for a free Okta Developer account. Log in to your developer console to get the following information.

Create a file named .env that has the following variables, all obtained from your Okta developer console:

• ISSUER

Log in to your developer console and navigate to API > Authorization Servers. Copy the Issuer URI for the default server.

SCOPE

Click on the word default to get details about the authorization server. Go to the **Scopes** tab and click the **Add Scope** button. Give it a name and optionally a description. The example app is for a parts manager, so for example you could name it parts_manager.

• CLIENT_ID

Navigate to **Applications**, then click the **Add Application** button. Select **Service**, then click **Next**. Choose a name then click **Done**. The **Client ID** is shown on the next page.

• CLIENT_SECRET

The Client Secret is on the same page as the Client ID

When you're done, your .env file should look something like this:

ISSUER=https://dev-123456.oktapreview.com/oauth2/default SCOPE=parts_manager CLIENT_ID=0123456789abcdefghij CLIENT_SECRET=0123456789abcdefghijklmnopqrstuvwxyz0123

Run the Server

To run the server, run npm start from the terminal.

Run the Client

To make secure API requests, you'll need to use the client located at client.js.

USAGE

node client url [method] [jsonString]

- url @er ùired# the path to your server along with the endpoint (e.g. http://localhost:3000/parts)
- method @pṛtiönal# the HTTP verb for the REST call (e.g. delete or post). Defaults to get
- jsonString @pṭiönal# stringified JSON data for use in put or post calls (e.g. '{"partNumber":"asdf-1234"}')

EXAMPLES

- node client http://localhost:3000/parts:get a list of all parts
- node client http://localhost:3000/parts post '{"partNumber":"asdf-1234"}' : creates a new part with part number
 asdf-1234
- node client http://localhost:3000/parts/1 : gets details about the part with and id of 1
- node client http://localhost:3000/parts/7 put '{"name":"A single dairy farm"}' : updates the name field of the part with an id of 7
- node client http://localhost:3000/parts/11 delete : deletes the part with an id of 11

License

Apache 2.0, see LICENSE.