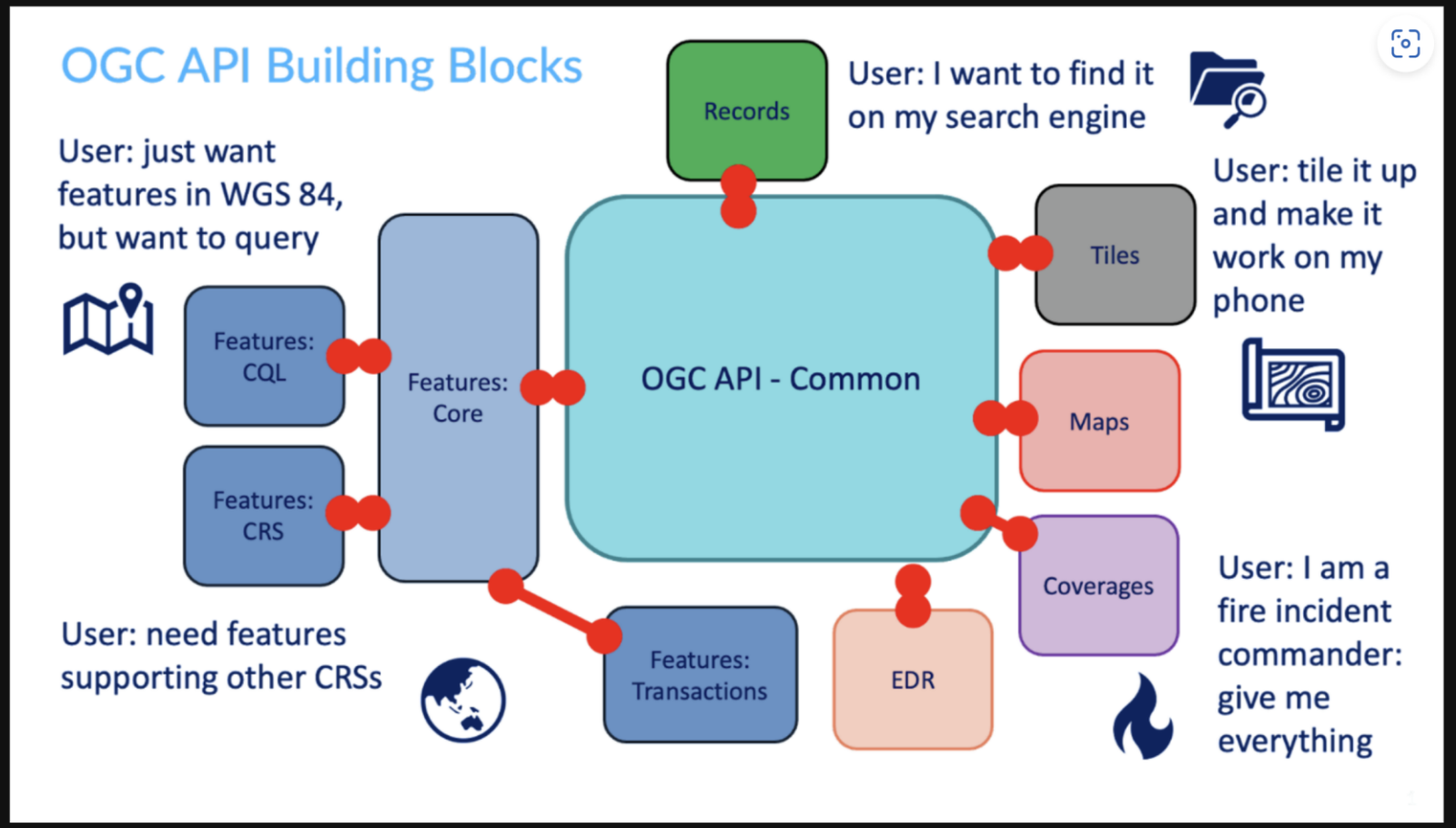
pygeoapi

Python implementation of OGC API standards

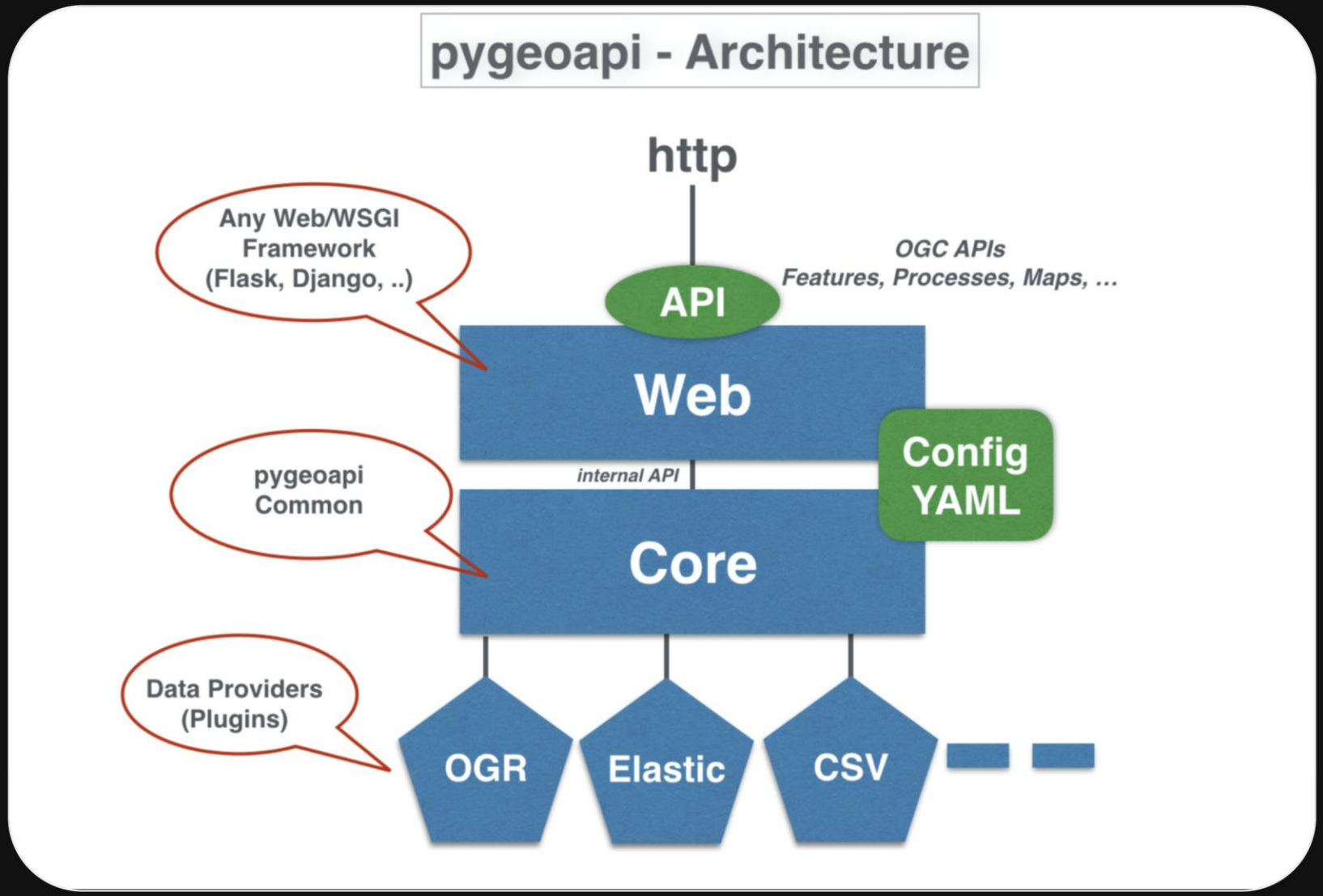
Krishna Lodha

krishnaglodha.com

OGC APIs Architecture

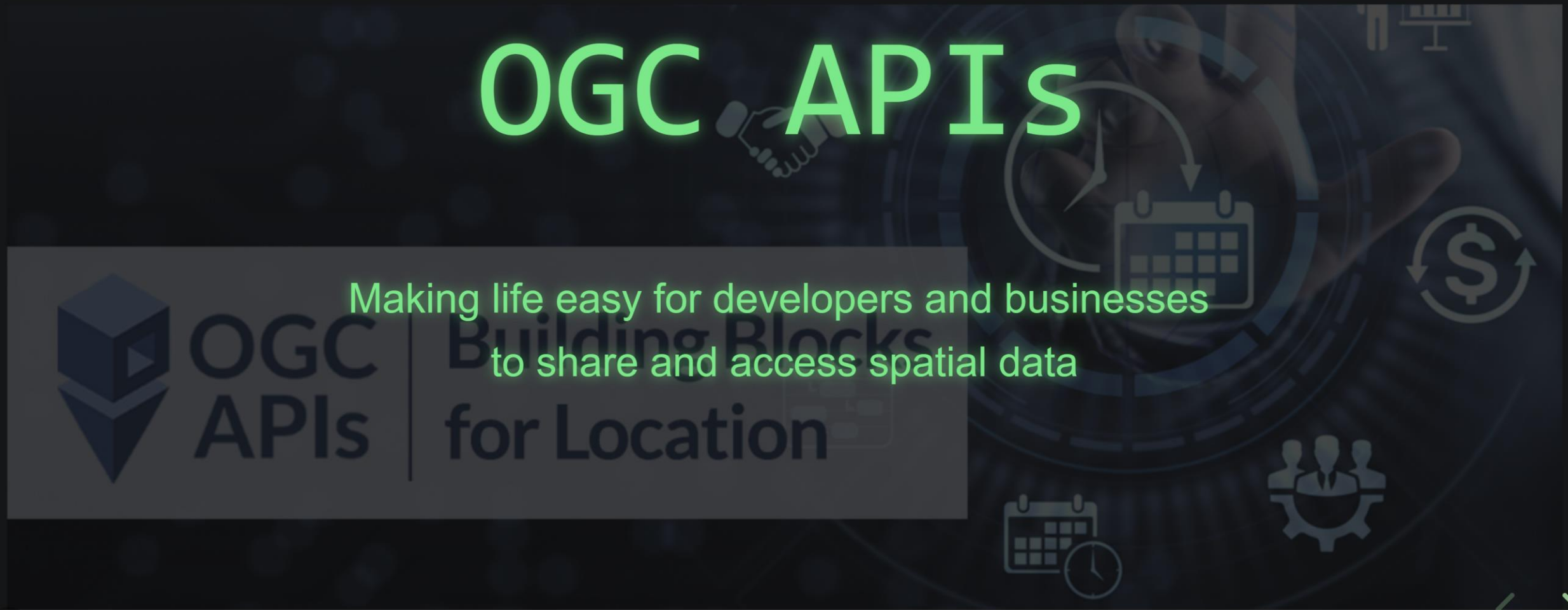


pygeoapi Architecture




2
Installation of pygeoapi and current state
4
OGC API Implementation
1
OGC APIs standards
3
Web Interface walkthrough and Config
5
Production Settings

OGC APIs



Making life easy for developers and businesses to share and access spatial data

Legacy Standards

WMS **WFS** **WCS**

WPS **WMTS**

https://www.ogc.org/docs/is

OGC APIs Standard Suite

Map

Tile

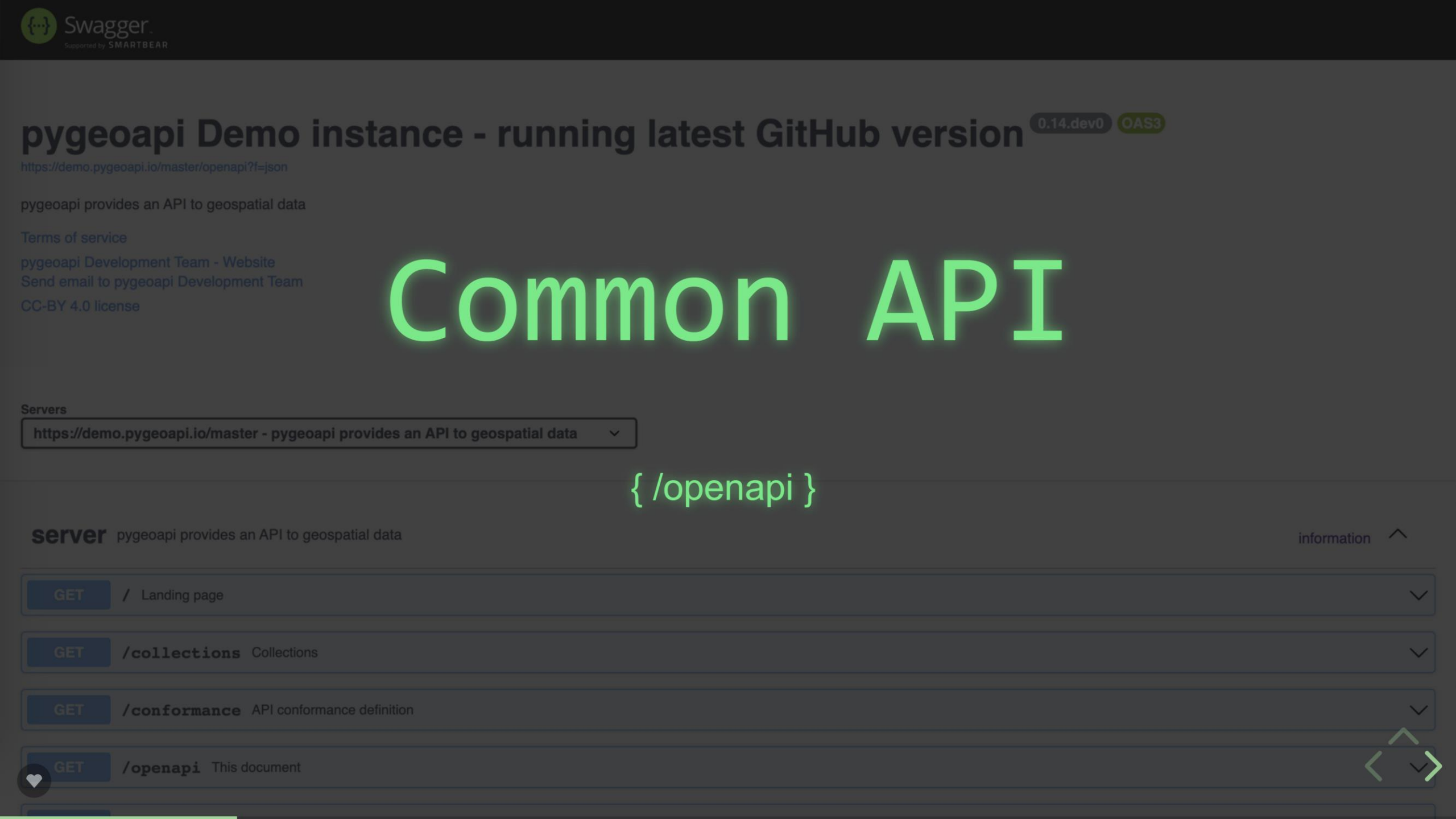
Features

Coverages

Processes

Records

https://ogcapi.ogc.org/


Swagger
Supported by SMARTBEAR
pygeoapi Demo instance - running latest GitHub version
0.14.dev0
OAS3
https://demo.pygeoapi.io/master/openapi?f=json
pygeoapi provides an API to geospatial data
Terms of service
pygeoapi Development Team - Website
Send email to pygeoapi Development Team
CC-BY 4.0 license
Common API
Servers
https://demo.pygeoapi.io/master - pygeoapi provides an API to geospatial data
{ /openapi }
server pygeoapi provides an API to geospatial data
information
GET / Landing page
GET /collections Collections
GET /conformance API conformance definition
GET /openapi This document

Installation

GIT

Docker

PyPi

Conda

Install in 5 minutes

```
python -m venv pygeoapi
cd pygeoapi
. bin/activate
git clone https://github.com/geopython/pygeoapi.git
cd pygeoapi
pip install --upgrade pip
pip install -r requirements.txt
python setup.py install
cp pygeoapi-config.yml example-config.yml
vi example-config.yml  # edit as required
export PYGEOAPI_CONFIG=example-config.yml
export PYGEOAPI_OPENAPI=example-openapi.yml
pygeoapi openapi generate $PYGEOAPI_CONFIG > $PYGEOAPI_OPENAPI
pygeoapi serve
curl http://localhost:5000
```

Pull latest Image from dockerhub

```
1 docker pull geopython/pygeoapi:latest
```

Get package from pypi

```
1 pip install pygeoapi
```

Get package from conda

```
1 conda install -c conda-forge pygeoapi
```

Walkthrough

Understanding code base for pygeoapi

Server configuration

Information about server and app

COPY

```
server:
  bind:
      host: 0.0.0.0  # listening address for incoming connections
      port: 5000  # listening port for incoming connections
  url: http://localhost:5000/  # url of server
  mimetype: application/json; charset=UTF-8  # default MIME type
  encoding: utf-8  # default server encoding
  language: en-US  # default server language
  gzip: false # default server config to gzip/compress responses to requests with gzip in the Accept-Encoding header
  cors: true  # boolean on whether server should support CORS
  pretty_print: true  # whether JSON responses should be pretty-printed
  limit: 10  # server limit on number of items to return

  templates: # optional configuration to specify a different set of templates for HTML pages. Recommend using absolute paths. Omit this to use the default provided templates
    path: /path/to/jinja2/templates/folder # path to templates folder containing the jinja2 template HTML files
    static: /path/to/static/folder # path to static folder containing css, js, images and other static files referenced by the template

  map:  # leaflet map setup for HTML pages
```

Logging configuration

Information about logging usage

```
logging:
    level: ERROR  # the logging level (see https://docs.python.org/3/library/logging.html#logging-levels)
    logfile: /path/to/pygeoapi.log  # the full file path to the logfile
```

Metadata configuration

Extra Information about company and app

```yaml
metadata:
    identification:
        title: pygeoapi default instance  # the title of the service
        description: pygeoapi provides an API to geospatial data  # some descriptive text about the service
        keywords:  # list of keywords about the service
            - geospatial
            - data
            - api
        keywords_type: theme  # keyword type as per the ISO 19115 MD_KeywordTypeCode codelist. Accepted values are discipline, temporal, place, theme, stratum
        terms_of_service: https://creativecommons.org/licenses/by/4.0/  # terms of service
        url: http://example.org  # informative URL about the service
    license:  # licensing details
        name: CC-BY 4.0 license
        url: https://creativecommons.org/licenses/by/4.0/
    provider:  # service provider details
        name: Organization Name
        url: https://pygeoapi.io
    contact:  # service contact details
```

Hiding Configuration

User can access the collection if they know name

```
resources:
    foo:
        title: my hidden resource
        visibility: hidden
```

Resource Config

Collection

Process

Stac-Collection

{Collection}

Add Various data stores in pygeoapi

- Feature
- Coverage
- Tile

CSV

```yaml
airports:
        type: collection
        title: All Airports in world
        description: The dataset shows the location of all airports in world.
        keywords:
            - airport
            - point data
        links:
            - type: text/csv
              rel: canonical
              title: data
              href: https://www.naturalearthdata.com/downloads/10m-cultural-vectors/airports/
              hreflang: en-US
        extents:
            spatial:
                bbox: [-180,-90,180,90]
                crs: http://www.opengis.net/def/crs/OGC/1.3/CRS84
        providers:
            - type: feature
              name: CSV
              data: /Users/krishnaglodha/Documents/learning/pygeoapi/pygeoapi/data/airports.csv
              id_field: ne_id
```

GeoJSON

COPY

```
Rail Networks 4326:
        type: collection
        title: All Rail Networks in USA 4326
        description: The dataset shows lines of all Rail Networks 4326 in USA.
        keywords:
            - Rail Networks
            - line data
        links:
            - type: text/csv
              rel: canonical
              title: data
              href: https://www.naturalearthdata.com/downloads/10m-cultural-vectors/airports/
              hreflang: en-US
        extents:
            spatial:
                bbox: [-180,-90,180,90]
                crs: http://www.opengis.net/def/crs/OGC/1.3/CRS84
        providers:
            - type: feature
              name: GeoJSON
              data: /Users/krishnaglodha/Documents/learning/pygeoapi/pygeoapi/data/road_4326.geojson
              id_field: uident
```

ESRI Feature Service

COPY

```
Property Setback:
        type: collection
        title: Property setback in michigan
        description: The dataset shows Polygons of Property setback in michigan.
        keywords:
            - Property setback
            - michigan
        links:
            - type: text/csv
              rel: canonical
              title: data
              href: https://www.naturalearthdata.com/downloads/10m-cultural-vectors/airports/
              hreflang: en-US
        extents:
            spatial:
                bbox: [-180,-90,180,90]
                crs: http://www.opengis.net/def/crs/OGC/1.3/CRS84
        providers:
            - type: feature
              name: ESRI
              data: https://portal1-geo.sabu.mtu.edu/server/rest/services/Hosted/100ft_Property_Setback/FeatureServer/0
```

Geopackage

COPY

```
All Countries:
        type: collection
        title: All Countries in world
        description: The dataset shows Polygons of All Countries in world.
        keywords:
            - All Countries
            - world
        links:
            - type: text/csv
              rel: canonical
              title: data
              href: https://www.naturalearthdata.com/downloads/10m-cultural-vectors/10m-admin-0-countries/
              hreflang: en-US
        extents:
            spatial:
                bbox: [-180,-90,180,90]
                crs: http://www.opengis.net/def/crs/OGC/1.3/CRS84
        providers:
            - type: feature
              name: OGR
              data:
                  source_type: GPKG
                  source:
```

PostGIS

COPY

```
Popular places from database:
        type: collection
        title: Popular places from database in world
        description: The dataset shows Points of Popular places from Geopackage file.
        keywords:
            - Popular places from database
            - world
        links:
            - type: text/csv
              rel: canonical
              title: data
              href: https://www.naturalearthdata.com/downloads/10m-cultural-vectors/airports/
              hreflang: en-US
        extents:
            spatial:
                bbox: [-180,-90,180,90]
                crs: http://www.opengis.net/def/crs/OGC/1.3/CRS84
        providers:
            - type: feature
              name: PostgreSQL
              data:
                  host: 127.0.0.1
                  port: 5432 # Default 5432 if not provided
```

ESRI Shapefile

COPY

```
Popular Places:
        type: collection
        title: Popular Places in world
        description: The dataset shows Points of Popular Places in michigan.
        keywords:
            - Popular Places
            - world
        links:
            - type: text/csv
              rel: canonical
              title: data
              href: https://www.naturalearthdata.com/downloads/10m-cultural-vectors/airports/
              hreflang: en-US
        extents:
            spatial:
                bbox: [-180,-90,180,90]
                crs: http://www.opengis.net/def/crs/OGC/1.3/CRS84
        providers:
            - type: feature
              name: OGR
              data:
                  source_type: ESRI Shapefile
                  source:
```

GeoTiFF

COPY

```
world-tiff:
        type: collection
        title: Natural World Tiff
        description: Natural World Tiff of resolution 1:10m
        keywords:
            - Natural World
            - Tiff
        links:
            - type: text/html
              rel: canonical
              title: Natural World Tiff
              href: https://www.naturalearthdata.com/downloads/10m-natural-earth-1/10m-natural-earth/
              hreflang: it
        extents:
            spatial:
                bbox: [-180,-90,180,90]
                crs: http://www.opengis.net/def/crs/OGC/1.3/CRS84
        providers:
            - type: coverage
              name: rasterio
              data: /Users/krishnaglodha/Documents/learning/pygeoapi/pygeoapi/data/NE1_LR_LC/NE1_LR_LC.tif # place correct path here
```

Vector Tile

Create Vector Tiles from Data using tippecanoe

```
tippecanoe --output-to-directory=tiles/ --force --minimum-zoom=1 --maximum-zoom=6 --drop-densest-as-needed --extend-zooms-if-still-dropping --no-tile-compression state_province.geojson
```

COPY

```
States:
        type: collection
        title: States in World
        description: All 4500 states in world
        keywords:
            - states
            - world
        links:
            - type: text/html
              rel: canonical
              title: information
              href: https://www.naturalearthdata.com/downloads/10m-cultural-vectors/10m-admin-1-states-provinces/
              hreflang: en-US
        extents:
            spatial:
                bbox: [-180 -90 180 90]
```

CQL-Query

CQL Query works on PostgreSQL and Elasticsearch (as of now!!)

```
1 http://localhost:4501/collections/Popular%20places%20from%20database/items?f=json&filter=WIT
```

CQL Query works on PostgreSQL and Elasticsearch (as of now!!)

```
1 URL - http://localhost:4501/collections/Popular%20places%20from%20database/items?f=json&filter
2 Type - POST
3 Body - { "eq":[{"property": "nameascii"},"Sotouboua"]}
```

Create Your OWN Format

Create file - pygeoapi/pygeoapi/provider/csv_wkt_.py

```
# =================================================================
#
# Authors: Krishna Lodha <krishnaglodha@gmail.com>
#
# Copyright (c) 2023  Krishna Lodha
#
# Permission is hereby granted, free of charge, to any person
# obtaining a copy of this software and associated documentation
# files (the "Software"), to deal in the Software without
# restriction, including without limitation the rights to use,
# copy, modify, merge, publish, distribute, sublicense, and/or sell
# copies of the Software, and to permit persons to whom the
# Software is furnished to do so, subject to the following
# conditions:
#
# The above copyright notice and this permission notice shall be
# included in all copies or substantial portions of the Software.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
# EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES
```

Create Your OWN Format

Add reference in plugin.py

```
PLUGINS = {
    'provider': {
        ...
         'CSVWKT': 'pygeoapi.provider.csv_wkt_.CSVWKTProvider',
         ....
    }
}
```

Build project again

COPY

```
python3 setup.py install
```

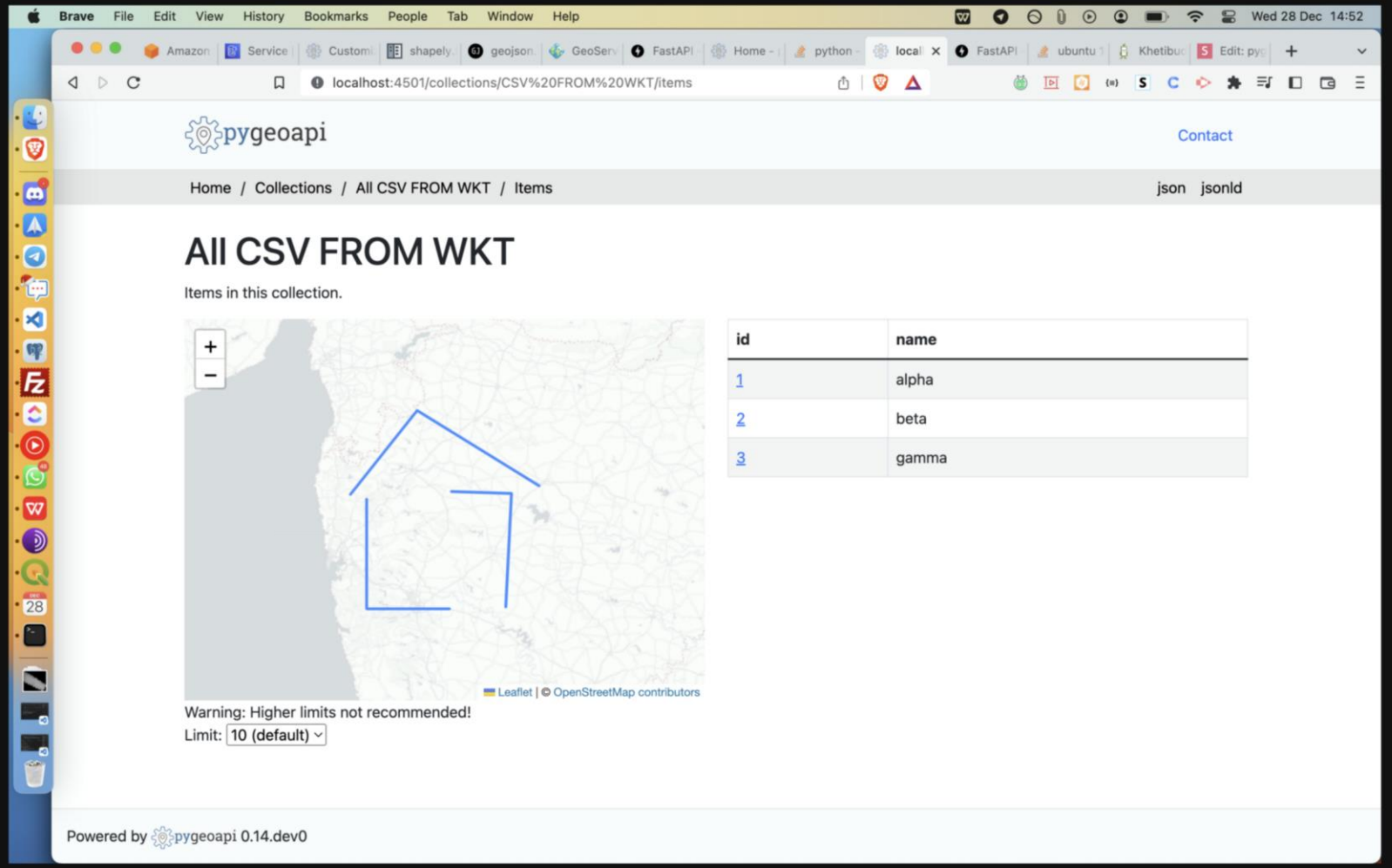
Create Your OWN Format

Add reference in config-file

```
CSV FROM WKT:
        type: collection
        title: All CSV FROM WKT
        description: The dataset shows the location of all CSV FROM WKT.
        keywords:
            - airport
            - point data
```

Create Open API File again

```
pygeoapi openapi generate $PYGEOAPI_CONFIG --output-file $PYGEOAPI_OPENAPI
```


Brave File Edit View History Bookmarks People Tab Window Help
Wed 28 Dec 14:52
localhost:4501/collections/CSV%20FROM%20WKT/items
pygeoapi
Contact
Home / Collections / All CSV FROM WKT / Items
json jsonld
All CSV FROM WKT
Items in this collection.
Leaflet | © OpenStreetMap contributors
Warning: Higher limits not recommended!
Limit: 10 (default)
id
name
1
alpha
2
beta
3
gamma
Powered by pygeoapi 0.14.dev0

{Process}

Share Python functions with OGC Process standards

OGC API - Processes

The OGC API - Processes enables the execution of computing processes and the retrieval of metadata describing their purpose and functionality. Typically, these processes combine raster, vector, and/or coverage data with well-defined algorithms to produce new raster, vector, and/or coverage information.

```
GET /processes
```

Lists the processes this API offers.

```
GET /processes/{process-id}
```

Returns a detailed description of a process.

```
GET /jobs
```

Returns the running and finished jobs for a process (optional).

```
POST /processes/{process-id}/execution
```

Executes a process, i.e. creates a new job. Inputs, outputs and the process id will have to be specified in a JSON document that needs to be send in the POST body.

```
GET /jobs/{job-id}
```

Returns the status of a job of a process.

```
DELETE /jobs/{job-id}
```

Cancel a job execution.

```
GET /jobs/{job-id}/results
```

Creating Sample Process

COPY

```
1
2  import logging
3
4  from pygeoapi.process.base import BaseProcessor, ProcessorExecuteError
5
6
7  LOGGER = logging.getLogger(__name__)
8
9  #: Process metadata and description
10 PROCESS_METADATA = {
11     'version': '0.2.0',
12     'id': 'hello-world',
13     'title': {
14         'en': 'Hello World',
15         'fr': 'Bonjour le Monde'
16     },
17     'description': {
18         'en': 'An example process that takes a name as input, and echoes '
19               'it back as output. Intended to demonstrate a simple '
20               'process with a single literal input.',
21         'fr': 'Un exemple de processus qui prend un nom en entrée et le '
22               'renvoie en sortie. Destiné à démontrer un processus '
23               'simple avec une seule entrée littérale.',
24     },
```

Adding Process to pygeoapi

1.

Create

Create process file and put it in process folder in pygeoapi

2.

Build

Rebuild pygeoapi package using

`python3 setup.py install`

3.

Launch

Mention Process name and path in config file

Making Async Requests

1.

Create

Manager to store processes

2.

Build

Jobs are created based on process execution

`/jobs`

3.

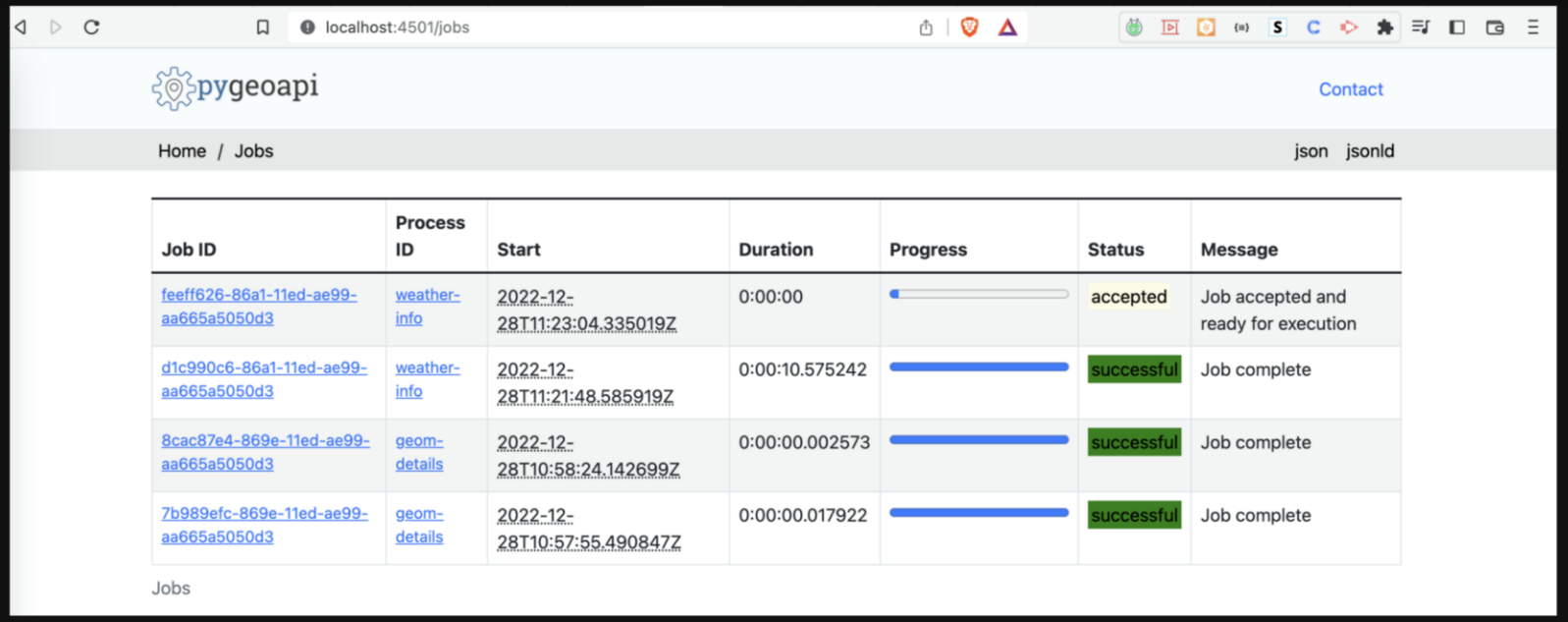
Launch

Results can be launched

`jobs/:id/results?f=json`

Add Manager

```yaml
server:
    manager:
        name: TinyDB
        connection: /Users/krishnaglodha/Documents/learning/pygeoapi/pygeoapi/data/pygeoapi-
        output_dir: /Users/krishnaglodha/Documents/learning/pygeoapi/pygeoapi/data/
```

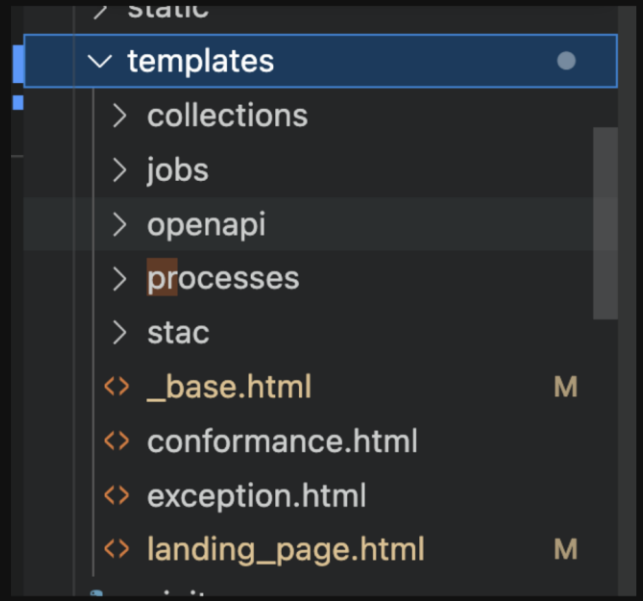


{Templates}

Create Custom pygeoapi webapp

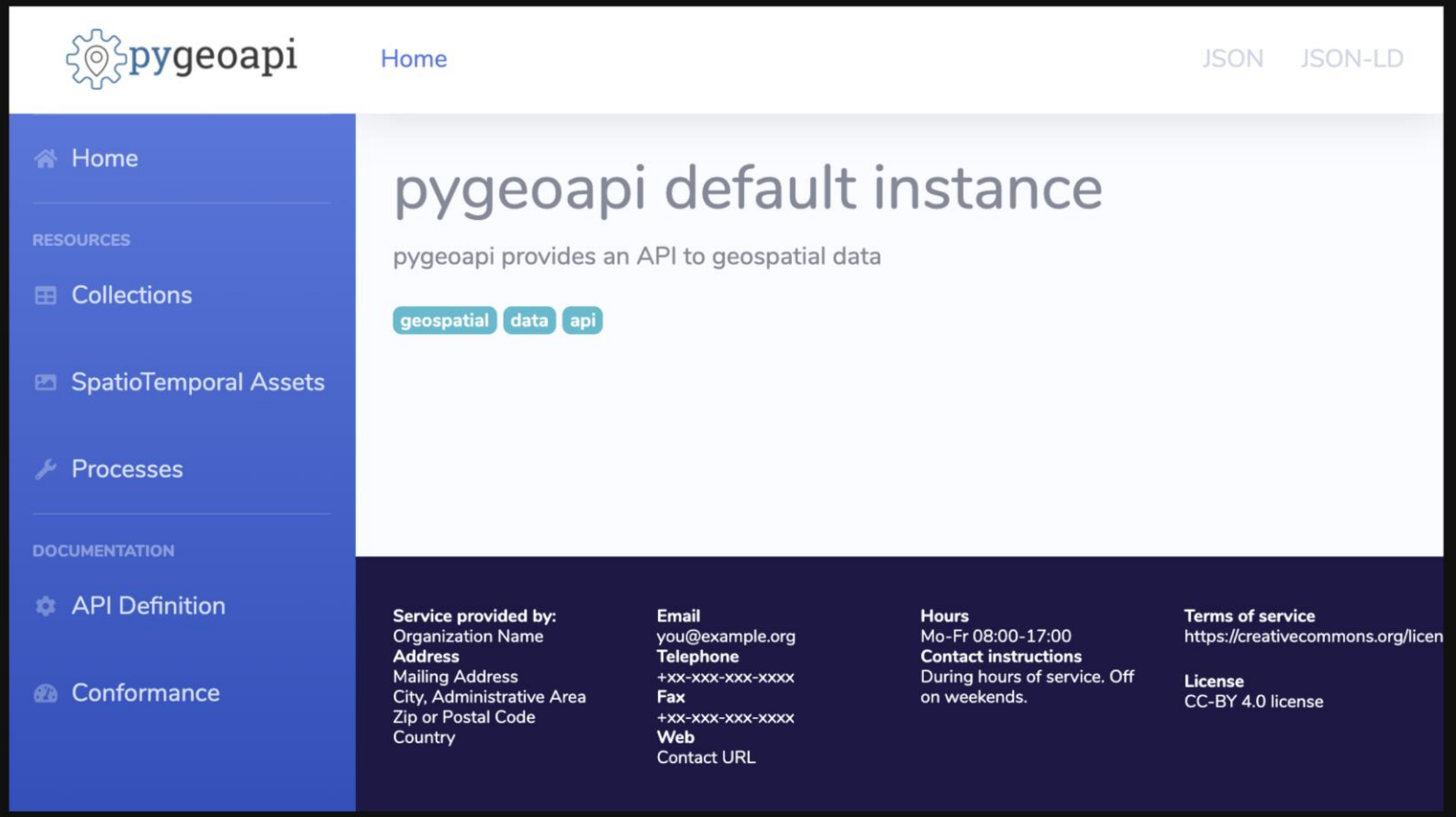
Try Default template folder

```
templates:
      path: /Users/krishnaglodha/Documents/learning/pygeoapi/pygeoapi/pygeoapi/templates
      static: /Users/krishnaglodha/Documents/learning/pygeoapi/pygeoapi/pygeoapi/static
```



Create your own templates

https://github.com/GeoCat/pygeoapi-skin-dashboard

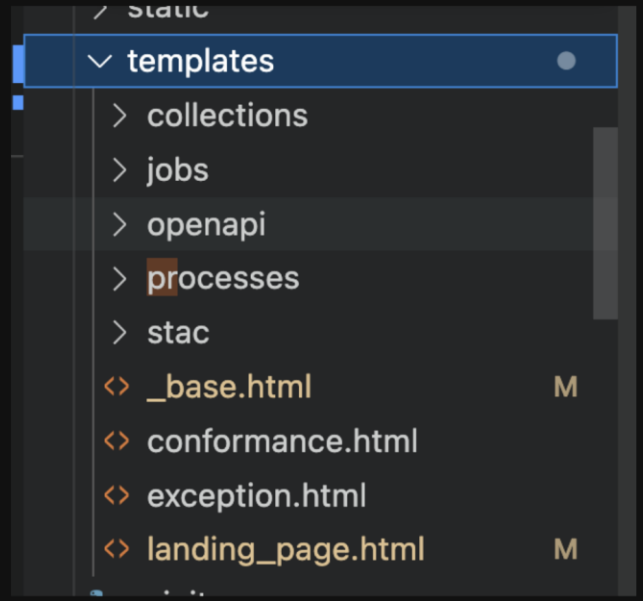


{Application}

Using pygeoapi in various ways

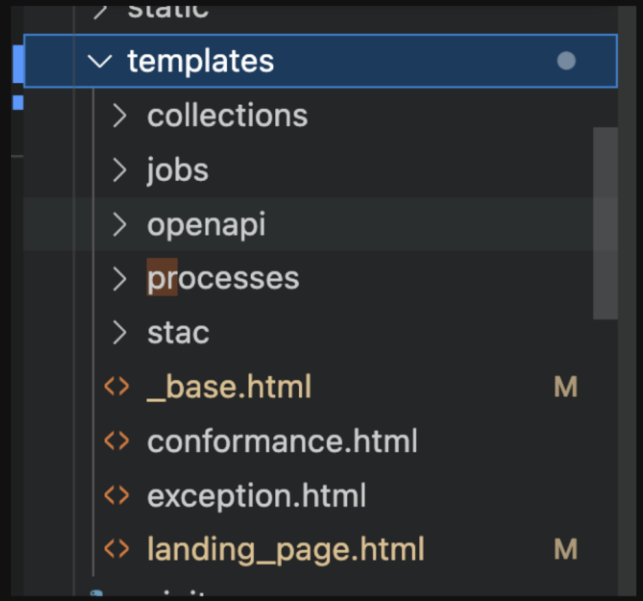
Try Default template folder

```
templates:
      path: /Users/krishnaglodha/Documents/learning/pygeoapi/pygeoapi/pygeoapi/templates
      static: /Users/krishnaglodha/Documents/learning/pygeoapi/pygeoapi/pygeoapi/static
```



Try Default template folder

```
templates:
      path: /Users/krishnaglodha/Documents/learning/pygeoapi/pygeoapi/pygeoapi/templates
      static: /Users/krishnaglodha/Documents/learning/pygeoapi/pygeoapi/pygeoapi/static
```



{owslib}

Py package following OGC Standards

Loading Core Features

```python
from owslib.ogcapi.features import Features

w = Features('http://localhost:4501/')
col = w.collections() ## Get all collections
feature_collections = w.feature_collections() # Get Feature collection
feature_collections
countries = w.collection_items('All Countries') # get Geojson of collection
countries
```

Output exceeds the size limit. Open the full output data in a text editor


```
{'type': 'FeatureCollection',
 'features': [{'type': 'Feature',
   'geometry': {'type': 'MultiPolygon',
    'coordinates': [[[[117.70360790395524, 4.163414542001791],
       [117.70360790400008, 4.163414542000055],
       [117.73807146200011, 4.157241908000032],
       [117.78357224700005, 4.157241908000032],
       [117.85255730900013, 4.157241908000032],
       [117.9070390276059, 4.156683015004099],
       [117.91211998800009, 4.144924221000053],
       [117.91822350400003, 4.100043036000045],
       [117.93482506600003, 4.059881903000075],
       [117.90137780000009, 4.036688544000071],
       [117.887054884, 4.031927802000041],
       [117.87256920700008, 4.03156159100007],
       [117.83855228000004, 4.040187893000052],
       [117.81910241000003, 4.06867096600007],
       [117.76270592500009, 4.100734768000052],
```