Annso Documentation

Release 1.0.0

Regovar Team

May 29, 2017

Contents

Contents:

CHAPTER 1

Quick guide

Deploye and use Annso in 5 minutes. In the below tutorial :

- <HOST> : is the server host, by example "www.annso.com"
- <PORT> ! is the port that will be use by the annso python application, by example 8080
- <ANNSO_PATH> : is the path on the server where is deployed the pirus python application, by example "/var/annso_v1"

Installation

The following tutorial will show you how to set up a quick development environment for the annso application on a linux server. You may need to install first

```
sudo apt install build-essential libssl-dev libffi-dev python3-dev virtualenv libpq-
↪dev
```

Annso need a postgresql database (9.5+). As ususal, you can customise value, just don't forget to update the config.py file accordingly

```
sudo apt install postgresql
psql -U postgres -c "CREATE USER annso WITH PASSWORD 'annso';"
psql -U postgres -c "DROP DATABASE IF EXISTS annso;"
psql -U postgres -c "CREATE DATABASE annso;"
psql -U postgres -c "GRANT ALL PRIVILEGES ON DATABASE annso to annso;"
```

Then clone the repository and install requirements

```
git clone https://github.com/REGOVAR/Annso.git
cd Annso
virtualenv -p /usr/bin/python3.5 venv
source venv/bin/activate
pip install -r requirements.txt
```

You will need to create following empty folder in the /var directory (you can change the location, but don’t forget to update the config.py file)

```
mkdir -p /var/regovar/annso
mkdir /var/regovar/annso/cache
mkdir /var/regovar/annso/downloads
mkdir /var/regovar/annso/files
```

Init database

```
psql -U annso -d annso -f <ANNSO_PATH>/annso/database/create_all.sql
psql -U annso -d annso -f <ANNSO_PATH>/annso/database/scripts/import_refgen.sql
```

Using NginX

Create the file into */etc/nginx/sites-available/annso* with the following content

Replace <PORT> and <HOST> with the good value:

```
#
# Virtual Host configuration for pirus.absolumentg.fr
#
upstream aiohttp_annso
{
        server 127.0.0.1:<PORT> fail_timeout=0;
}

server
{
        listen 80;
        listen [::]:80;
        server_name <HOST>;

        location / {
                # Need for websockets
                proxy_http_version 1.1;
                proxy_set_header Upgrade $http_upgrade;
                proxy_set_header Connection "upgrade";

                proxy_set_header Host $http_host;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_redirect off;
                proxy_buffering off;
                proxy_pass http://aiohttp_annso;
        }

        location /static {
                root /var/regovar/annso;
        }
}
```

Enable this virtual host by creating a symbolic link

```
sudo ln -s /etc/nginx/sites-enable/annso /etc/nginx/sites-available/annso
sudo /etc/init.d/nginx restart
```

Don’t forget to modify the <ANNSO_PATH>/annso/config.py file according to your configuration.

Run Annso

just

```
cd <ANNSO_PATH>/annso
make app
```

Using Annso

Create an analysis

todo

Setup samples

todo

Create and apply filters

todo

Select variant and get result

todo

CHAPTER 2

Developer Guide

Solution organisation

- **The core team of Annso project:**
  - As sub project of Revogar, the core team of Annso, is the same as for Regovar : Ikit, dridk, Oodnadatta and Arkanosis. All of them are both consultant and developer.
- **Coding Rules :**
  - https://www.python.org/dev/peps/pep-0008/
- **Git branching strategy :**
  - Dev on master,
  - One branch by release; with the version number as name (by example branch "v1.0.0" for the v1.0.0)
- **Discussion :**
  - https://regovar.slack.com/
  - dev@regovar.org

Architecture

See dedicated page

Model

Analyse

Static property :

public_fields <str[]> : liste des champs exportable pour le enduser (client annso)

Public properties :

id <int> : id of the sample in the database

name <str> : (required) name of the sample when imported (name in the vcf file by example)

comment <str> : user can add some comments about the sample

is_mosaic <bool> : true if the sample is [mosaic](https://www.wikiwand.com/en/Mosaic_(genetics)); false otherwithe

Internal properties :

-

Static methods :

from_id(pipe_id) : return a Pipeline object from the database

Internal methods :

export_server_data(self)

export_client_data(self)

import_data(self, data)

url(self) : return the url that shall be used to download the pipeline package

upload_url(self) : return the url that shall be used to upload the pipeline on the server

Sample

Static property :

public_fields <str[]> : liste des champs exportable pour le enduser (client annso)

Public properties :

id <int> : id of the sample in the database

name <str> : (required) name of the sample when imported (name in the vcf file by example)

comment <str> : user can add some comments about the sample

is_mosaic <bool> : true if the sample is [mosaic](https://www.wikiwand.com/en/Mosaic_(genetics)); false otherwithe

Internal properties :

-

Static methods :

from_id(pipe_id) : return a Pipeline object from the database

Internal methods :

export_server_data(self)

export_client_data(self)

import_data(self, data)

url(self) : return the url that shall be used to download the pipeline package

upload_url(self) : return the url that shall be used to upload the pipeline on the server

API

See dedicated page for the current api implemented.

- How to update current api
- Implement a new version of the api

TUS.IO protocol

CHAPTER 3

Indices and tables

- genindex
- modindex
- search