DOCUMENT 1

Installing the Comparative Pathology Workbench (CPW) in a Development Environment

This document describes how to set up and run the Software for the Comparative Pathology Workbench (CPW) in a DEVELOPMENT environment ONLY!

Mike Wicks
24th November 2023

Step 1 – Software Prerequisites

Install Postgres (greater than version 11) and create a database with a database user – you may find it helpful to install a Postgres Client like "pghAdmin4" to help you manage the database.

Pull the software from the Git repository:
https://github.com/Comparative-Pathology/comparativepathologyworkbench

Install "Conda" or "Microconda", software that runs a virtual environment and manages packages and environments.

Configure a local webserver to serve at a non-standard port (ie. Not port 80).

Step 2 – Set Up and Activate a Virtual Environment

Conda requires a "**environment.yml**" that specifies all the required Library dependencies.

Move to the "**app**" folder - Create a virtual environment:
conda env create --name test_env --file environment.yml

Start the virtual environment:
conda activate test_env

(You can deactivate the current environment, thus:
conda deactivate
You can list available environments, thus:
conda env list
You can delete environments, thus:
conda remove --name test_env –all)

Step 2 – Configure Software with a Database

The system needs a file to hold environment variables.

In the "**app/config**" sub-folder create a "**.env**" file with the following environment variables.

An example "**.env**" file is shown here:

```
SECRET_KEY=a_secret_key
ENCRYPT_KEY=an_encrypt_key
CPW_CIPHER_STRING=a_cpw_cipher_string
DEBUG=True
ALLOWED_HOSTS=localhost, 127.0.0.1
DB_ENGINE=django.db.backends.postgresql
DB_NAME=a_schema_name
DB_USER=a_database_user
```

```
DB_PASSWORD=a_password
DB_HOST=localhost
DB_PORT=5432
DB_ATOMIC_REQUESTS=True
EMAIL_BACKEND=django.core.mail.backends.filebased.EmailBackend
EMAIL_FILE_PATH=a_folder_somewhere
SESSION_EXPIRE_AT_BROWSER_CLOSE=True
SESSION_COOKIE_AGE=86400
```

Step 3 - Set up an Empty Database

Run all Django Migrations, to set up an empty database.

python manage.py migrate

Create Superuser Account

python manage.py createsuperuser

Step 4 - Set up Base Data in the Database

The following commands populate the database with the basic data required for the system to function.

python manage.py loaddata fixtures/protocol_prod.json --app matrices.protocol
python manage.py loaddata fixtures/type_prod.json --app matrices.type
python manage.py loaddata fixtures/location_prod.json --app matrices.location
python manage.py loaddata fixtures/command_prod.json --app matrices.command
python manage.py loaddata fixtures/blog_prod.json --app matrices.blog
python manage.py loaddata fixtures/environment_prod.json --app matrices.environment
python manage.py loaddata fixtures/authority_prod.json --app matrices.authority
python manage.py loaddata fixtures/collectionauthority_prod.json --app matrices.collectionauthority
python manage.py loaddata fixtures/server_prod.json --app matrices.server

Step 5 – Set up Database Views

In **fixtures/views.sql** there is a set of SQL statements that create the views required by the application.

Run these SQL statements using your chosen Postgres client – You will need to tailor the GRANT statements to match your chosen database name and database user.

Step 6 – Run the System

python manage.py runserver

Point a Browser at http://localhost:8000

You should now see the CPW Home Page

Step 7 - POSTSCRIPT

However, the application CANNOT do anything until it has been linked to a WordPress instance.

See the document “CPW_Administration” for more details.