**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.