Set up a python virtual environment

Make sure ubuntu system version is recent enough

```
lsb_release -a
```

Output (or higher)

Distributor ID: Ubuntu

Description: Ubuntu 24.04.3 LTS

Release: 24.04

Codename: noble

```
python3 -V
```

Output

Python 3.12.3

Make a base for all our virtual environments

```
mkdir ${HOME}/.venv
# Now we can create all our virtual environments  under ${HOME}/.venv/
```

Make a Python Virtual Environment

```
# Here we call our first environment devpi
cd ${HOME}/.venv/
python3 -m venv devpi
```

OR using venvwrapper

```
mkvenv devpi
```

Once we have our environment created. We can install the elements we need to get started.

Activate It

```
source ${HOME}/.venv/devpi/bin/activate
```

Or with venvwrapper

```
venv devpi
```

install the basics

```
# install
python -m pip install -U pip
pip install wheel
pip install setuptools
pip install twine
pip install rst2pdf
pip install flit

# de-activate
deactivate
```

Saving prerequisites

Once your virtual environment is complete, it is nice to save the context. Then you can quicky restore your virtual environment from scratch if need be.

While Activated

```
pip freeze >requirements.txt

# It may be a good idea to keep such files under version control.
```

Recovering While Activated

```
pip install -r requirements.txt
```

Automatic activation

In many cases we want to activate a project whenever we log in. We can do that automatically by adding something like this to .bashrc and/or .profile

```
#
# Added for automatic vitual environment activation
#
if [ -f "${HOME}/.venv/devpi/bin/activate" ] ; then
    source ${HOME}/.venv/devpi/bin/activate
fi
```

OR with venvwrapper

```
#
# Added for automatic vitual environment activation
#
VENV_HOME=${HOME}/.venv
venv_wrapper=$(which venvwrapper.sh)
if [[ -n $venv_wrapper ]]; then
    source $venv_wrapper
    venv devpi
fi
```