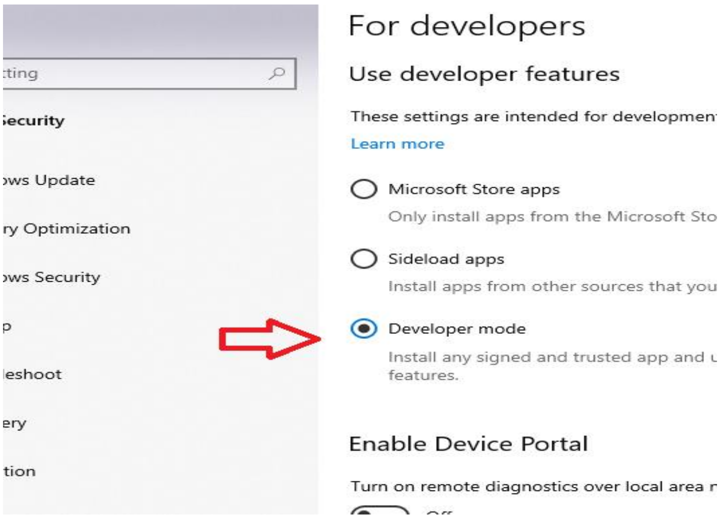
Installing apache airflow in windows

Step-1:

Enable developer mode in Windows in 'Developer settings'.



For developers

Use developer features

These settings are intended for developmen

Learn more

○ Microsoft Store apps
Only install apps from the Microsoft Sto

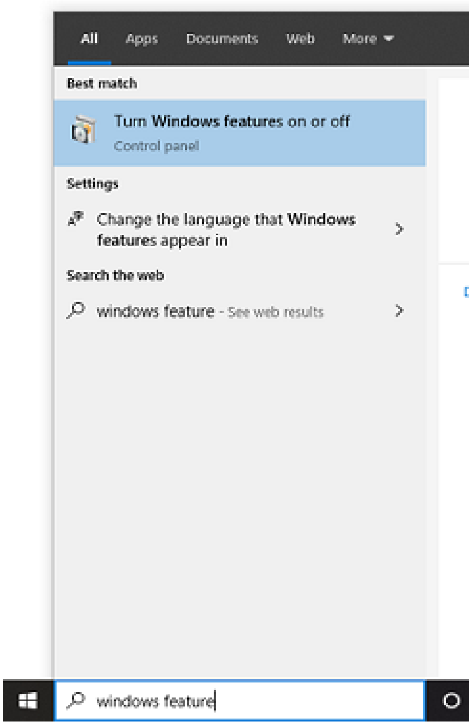
○ Sideload apps
Install apps from other sources that you

● Developer mode
Install any signed and trusted app and u
features.

Enable Device Portal

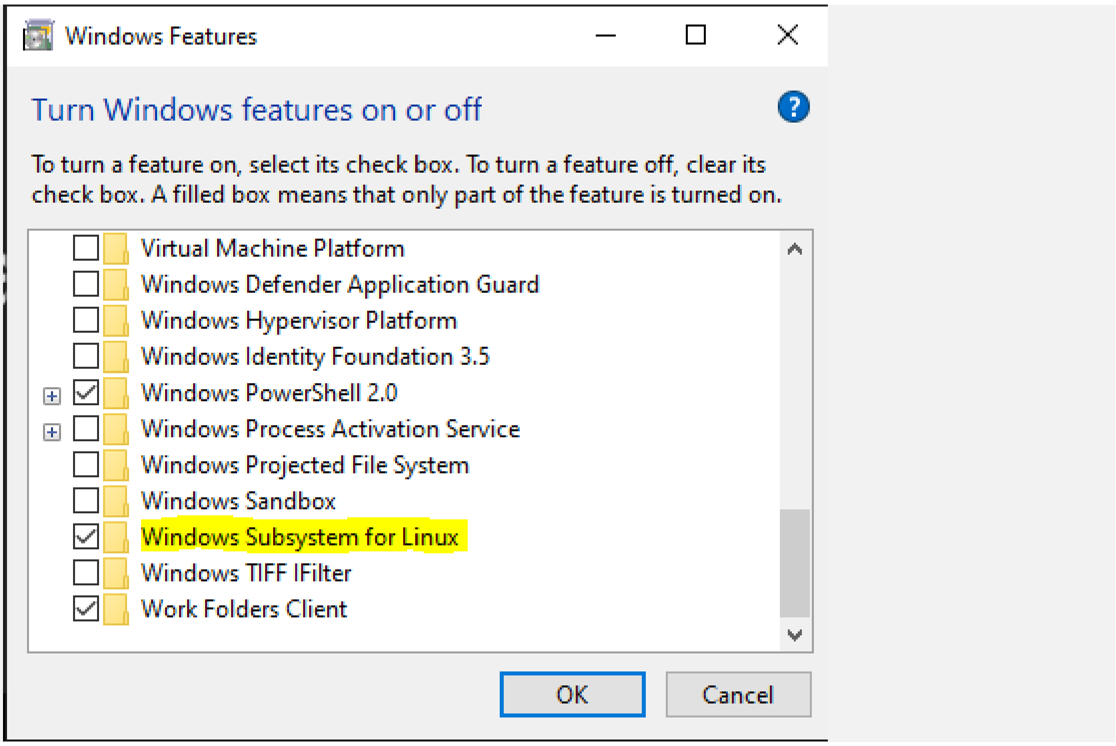
Turn on remote diagnostics over local area r

Enable 'Windows Subsystem for Linux' in 'Turn windows features on or off'.



All
Apps
Documents
Web
More
Best match
Turn Windows features on or off
Control panel
Settings
Change the language that Windows features appear in
Search the web
windows feature - See web results
windows feature

Check the box next to Windows Subsystem for Linux and click OK.



Windows Features

Turn Windows features on or off

To turn a feature on, select its check box. To turn a feature off, clear its check box. A filled box means that only part of the feature is turned on.

- ☐ Virtual Machine Platform
- ☐ Windows Defender Application Guard
- ☐ Windows Hypervisor Platform
- ☐ Windows Identity Foundation 3.5
- ⊞ ☑ Windows PowerShell 2.0
- ⊞ ☐ Windows Process Activation Service
- ☐ Windows Projected File System
- ☐ Windows Sandbox
- ☑ Windows Subsystem for Linux
- ☐ Windows TIFF IFilter
- ☑ Work Folders Client

Download C++ Build tools.
Link: https://visualstudio.microsoft.com/visual-cpp-build-tools/

Initial installation process:
Open ubuntu from the startup menu.
In the prompt, set a username and password.

Step 2: Type the following commands in linux terminal for Installing PIP

Note:I installed using python 2.1.17 if you have updated version enter python3 instead of python and pip3 instead of pip

```
>sudo apt-get install software-properties-common

>sudo apt-add-repository universe

>sudo apt-get update

#updates the repository package
```

```
>sudo apt-get install python-setuptools
```

```
>sudo apt install python-pip
```

```
>sudo -H pip install --upgrade pip
```

Step 3:Installing Airflow dependencies:

These commands ensure that all the required packages are installed and up-to-date.

```
>sudo apt-get install libmysqlclient-dev
```

```
>sudo apt-get install libssl-dev
```

```
>sudo apt-get install libkrb5-dev
```

```
>sudo apt-get install libsasl2-dev
```

Installing postgresql via CLI:

```
>sudo apt-get install postgresql postgresql-contrib
```

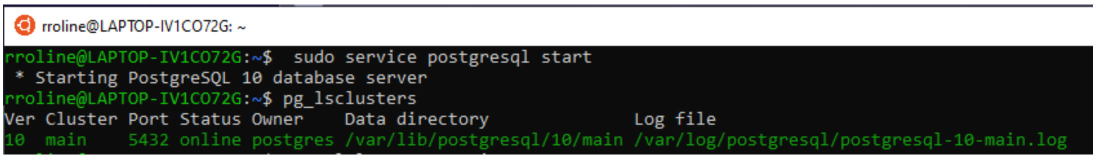
#This installs required psql packages, creates a cluster - 12(version) main(cluster)

#Enter the following command to start the postgresql service

```
>sudo service postgresql start
```

Check the status of the psql cluster to make sure it's running:

```
>pg_lsclusters
```

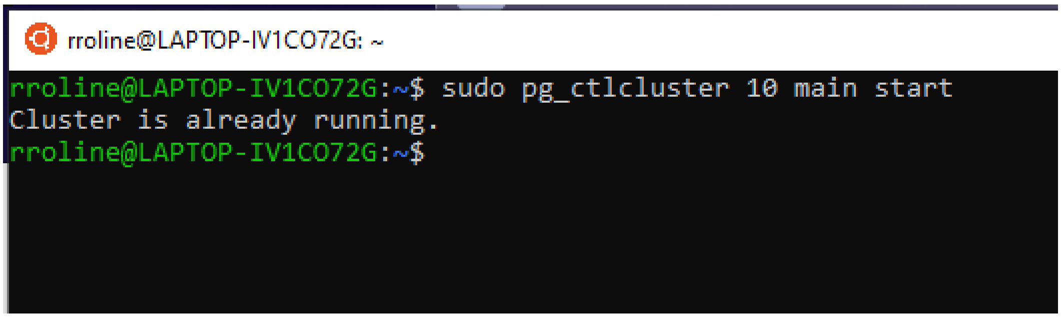


rroline@LAPTOP-IV1CO72G: ~

```
rroline@LAPTOP-IV1CO72G:~$  sudo service postgresql start
 * Starting PostgreSQL 10 database server
rroline@LAPTOP-IV1CO72G:~$ pg_lsclusters
Ver Cluster Port Status Owner    Data directory              Log file
10  main    5432 online postgres /var/lib/postgresql/10/main /var/log/postgresql/postgresql-10-main.log
```

Enter the version and cluster versions from the above output

```
>sudo pg_ctlcluster <version> <cluster> start
```



rroline@LAPTOP-IV1CO72G: ~

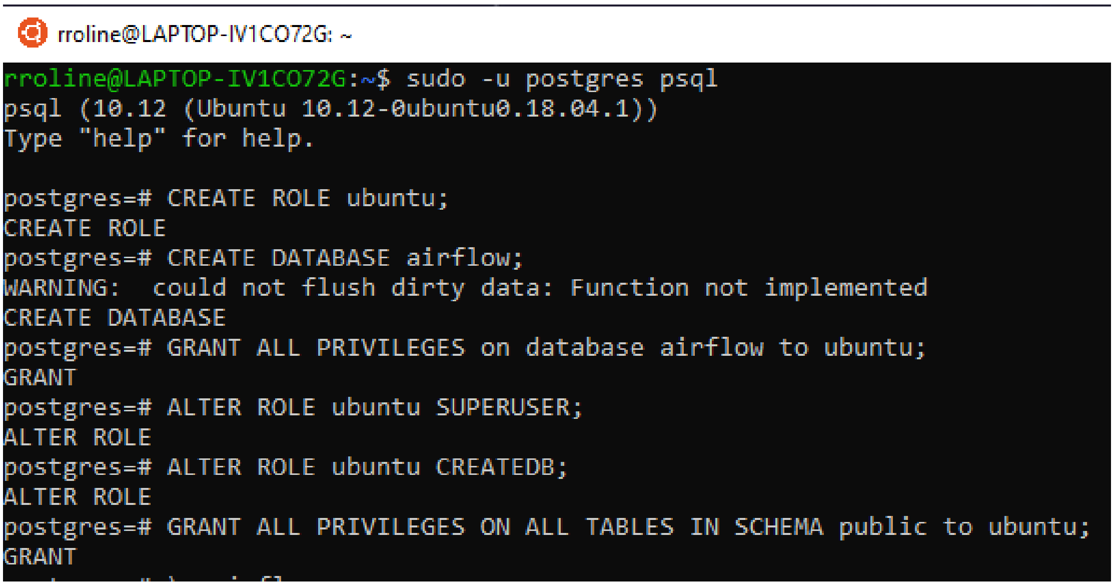
```
rroline@LAPTOP-IV1CO72G:~$ sudo pg_ctlcluster 10 main start
Cluster is already running.
rroline@LAPTOP-IV1CO72G:~$
```

Create a database to use for Airflow, create a profile and grant all privileges:

```
>sudo -u postgres psql
```

#gets access to psql

```
>>>CREATE ROLE ubuntu;
>>>CREATE DATABASE airflow;
>>>GRANT ALL PRIVILEGES on database airflow to ubuntu;
>>>ALTER ROLE ubuntu SUPERUSER;
>>>ALTER ROLE ubuntu CREATEDB;
>>>ALTER ROLE ubuntu LOGIN;
>>>GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public to ubuntu;
```

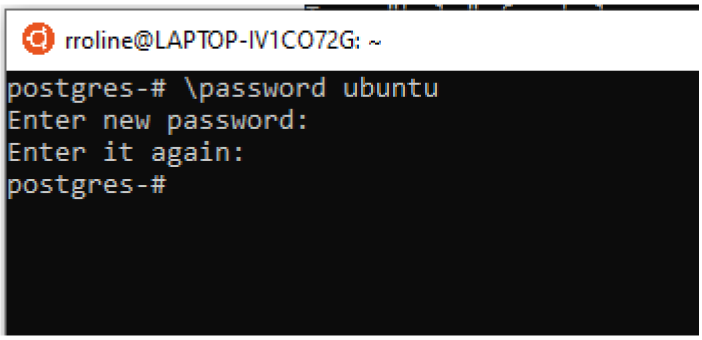


rroline@LAPTOP-IV1CO72G: ~

```
rroline@LAPTOP-IV1CO72G:~$ sudo -u postgres psql
psql (10.12 (Ubuntu 10.12-0ubuntu0.18.04.1))
Type "help" for help.

postgres=# CREATE ROLE ubuntu;
CREATE ROLE
postgres=# CREATE DATABASE airflow;
WARNING:  could not flush dirty data: Function not implemented
CREATE DATABASE
postgres=# GRANT ALL PRIVILEGES on database airflow to ubuntu;
GRANT
postgres=# ALTER ROLE ubuntu SUPERUSER;
ALTER ROLE
postgres=# ALTER ROLE ubuntu CREATEDB;
ALTER ROLE
postgres=# GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public to ubuntu;
GRANT
```

Setup a password for ubuntu user:

```
>>>\password ubuntu:
```



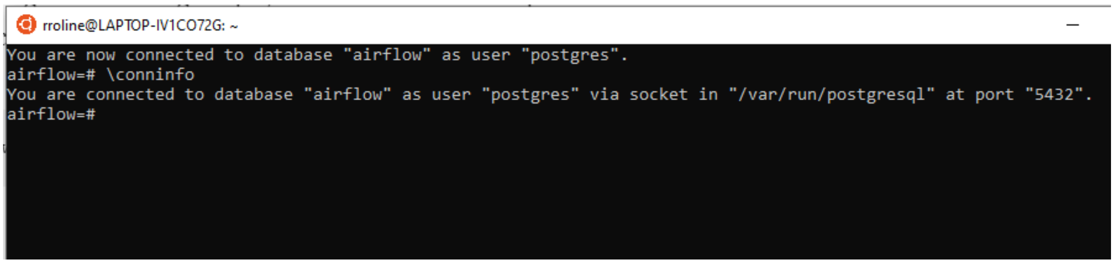
rroline@LAPTOP-IV1CO72G: ~

```
postgres-# \password ubuntu
Enter new password:
Enter it again:
postgres-#
```

```
Check the connection to airflow database:
>>>\c airflow
```

#You are now connected to database "airflow" as user "postgres".

```
airflow=#>>> \conninfo
```



rroline@LAPTOP-IV1CO72G: ~

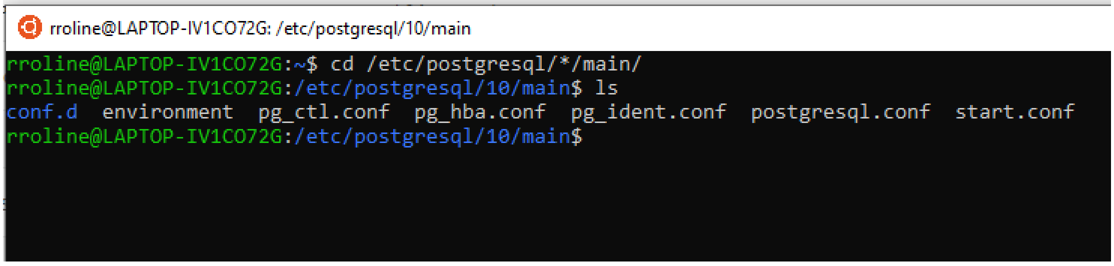
```
You are now connected to database "airflow" as user "postgres".
airflow=# \conninfo
You are connected to database "airflow" as user "postgres" via socket in "/var/run/postgresql" at port "5432".
airflow=#
```

#You are connected to database "airflow" as user "postgres" via socket in "/var/run/postgresql" at port "5432".
#Hit ctrl+Z to stop the session

Modifying values in config files(pb_hba.conf, postgresql.conf) in /etc/postgresql/*/main/ to finish airflow set up:

```
>cd /etc/postgresql/*/main/
```

```
>ls
```

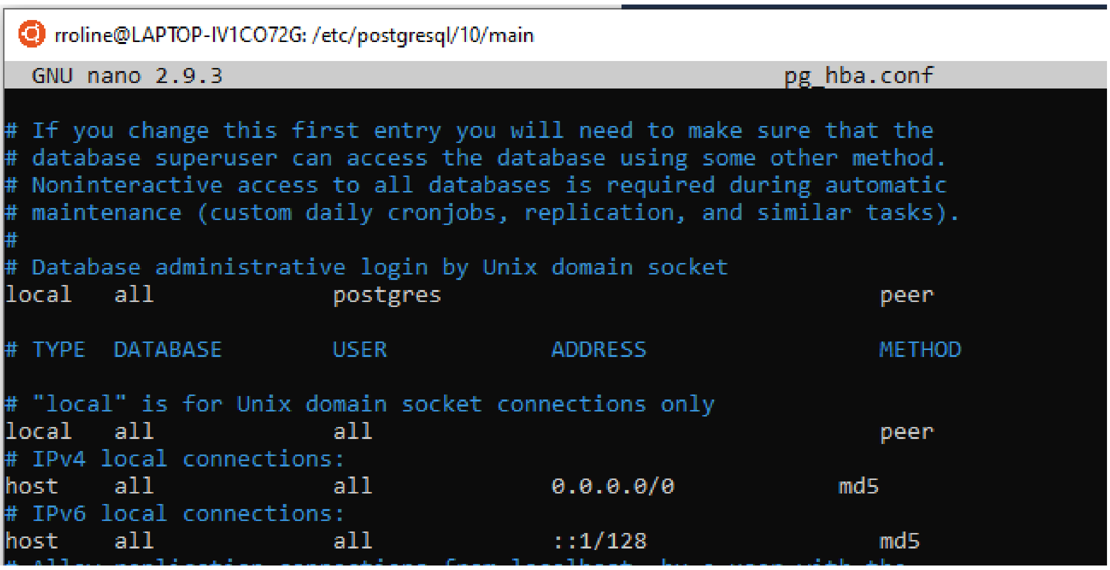


rroline@LAPTOP-IV1CO72G: /etc/postgresql/10/main

```
rroline@LAPTOP-IV1CO72G:~$ cd /etc/postgresql/*/main/
rroline@LAPTOP-IV1CO72G:/etc/postgresql/10/main$ ls
conf.d  environment  pg_ctl.conf  pg_hba.conf  pg_ident.conf  postgresql.conf  start.conf
rroline@LAPTOP-IV1CO72G:/etc/postgresql/10/main$
```

#conf.d environment pg_ctl.conf pg_hba.conf pg_ident.conf postgresql.conf start.conf

```
>sudo nano pg_hba.conf
```



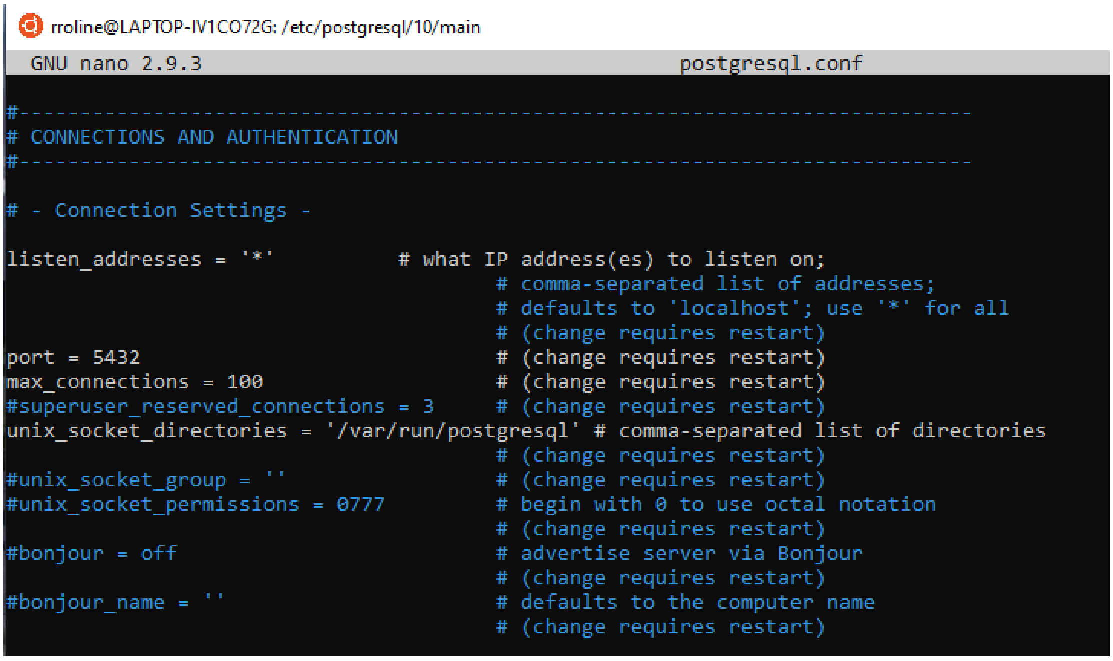
rroline@LAPTOP-IV1CO72G: /etc/postgresql/10/main

```
  GNU nano 2.9.3                                         pg_hba.conf

# If you change this first entry you will need to make sure that the
# database superuser can access the database using some other method.
# Noninteractive access to all databases is required during automatic
# maintenance (custom daily cronjobs, replication, and similar tasks).
#
# Database administrative login by Unix domain socket
local   all             postgres                                peer

# TYPE  DATABASE        USER            ADDRESS                 METHOD

# "local" is for Unix domain socket connections only
local   all             all                                     peer
# IPv4 local connections:
host    all             all             0.0.0.0/0            md5
# IPv6 local connections:
host    all             all             ::1/128                 md5
```

#Edit under IPv4 local connections to 0.0.0.0/0
#ctrl-s to save, ctrl-x to exit

```
>sudo nano postgresql.conf
```

#Edit listen_addresses = '*' (uncommenmt after modification)



```
rroline@LAPTOP-IV1CO72G: /etc/postgresql/10/main
  GNU nano 2.9.3                                    postgresql.conf

#------------------------------------------------------------------------------
# CONNECTIONS AND AUTHENTICATION
#------------------------------------------------------------------------------

# - Connection Settings -

listen_addresses = '*'          # what IP address(es) to listen on;
                                        # comma-separated list of addresses;
                                        # defaults to 'localhost'; use '*' for all
                                        # (change requires restart)
port = 5432                             # (change requires restart)
max_connections = 100                   # (change requires restart)
#superuser_reserved_connections = 3     # (change requires restart)
unix_socket_directories = '/var/run/postgresql' # comma-separated list of directories
                                        # (change requires restart)
#unix_socket_group = ''                 # (change requires restart)
#unix_socket_permissions = 0777         # begin with 0 to use octal notation
                                        # (change requires restart)
#bonjour = off                          # advertise server via Bonjour
                                        # (change requires restart)
#bonjour_name = ''                      # defaults to the computer name
                                        # (change requires restart)
```

```
#ctrl-s to save, ctrl-x to exit
```

Restart postgresql service to save and load changes:

```
>sudo service postgresql restart
```

#Allow firewall access if prompted

```
>cd ~
```

#goes back to root directory

Step 4: Installing Apache Airflow

```
>sudo SLUGIFY_USES_TEXT_UNIDECODE=yes pip install apache-airflow
Now, add path to PATH within terminal window allowing us to call airfow directly
using the airflow command:
This on executing in CLI only changes the PATH temporarily. To change PATH
permanently, add the above line at:
>sudo nano ~/.bashrc
Close the terminal and open a new instance of Ubuntu for further procedures.
```

Step 5: Apache Airflow Setup

Apache airflow setup:

```
>airflow db init
```

#initializes the database and creates necessary config files in the newly created airflow directory.

```
>cd airflow
```

Make necessary changes in the airflow config file as:

```
>sudo nano airflow.cfg
#dags_folder = /mnt/c/dags
#base_log_folder = /mnt/c/dags/logs
#executor = CeleryExecutor
#load_examples = False
#expose_config = True
#sql_alchemy_conn = postgresql+psycopg2://ubuntu:<password>@localhost:5432/airflow
#broker_url = amqp://guest:guest@localhost:5672//
#result_backend = amqp://guest:guest@localhost:5672//
#result_backend = amqp://guest:guest@localhost:5672//
Here, mnt/c points to C:/ in Windows.
```

Create dags and logs folders in this C:/ of windows.Then run the following to install psycopg2 (what has been mentioned in airflow.cfg):

```
>sudo apt-get update -y
```

```
>sudo apt-get install -y libpq-dev
```

```
>pip install psycopg2
```

RabbitMQ is a messaging broker, an inermediary for messaging. It gives applications a common patform to send and receive messages, and your messages a safe place to live until received.

Install Rabbitmq:

```
>sudo apt install rabbitmq-server
```

And update the rabbitmq config file:

```
>sudo nano /etc/rabbitmq/rabbitmq-env.conf
```

#change NODE_IP_ADDRESS=0.0.0.0

Start the RabbitMQ service:

```
>sudo service rabbitmq-server start
```

Install Celery:

```
>sudo pip install 'celery>=3.1.17,<4.0'
```

```
>airflow db intit
```

#runs again

Now we are ready to open the airflow webserver and scheduler.
Run:

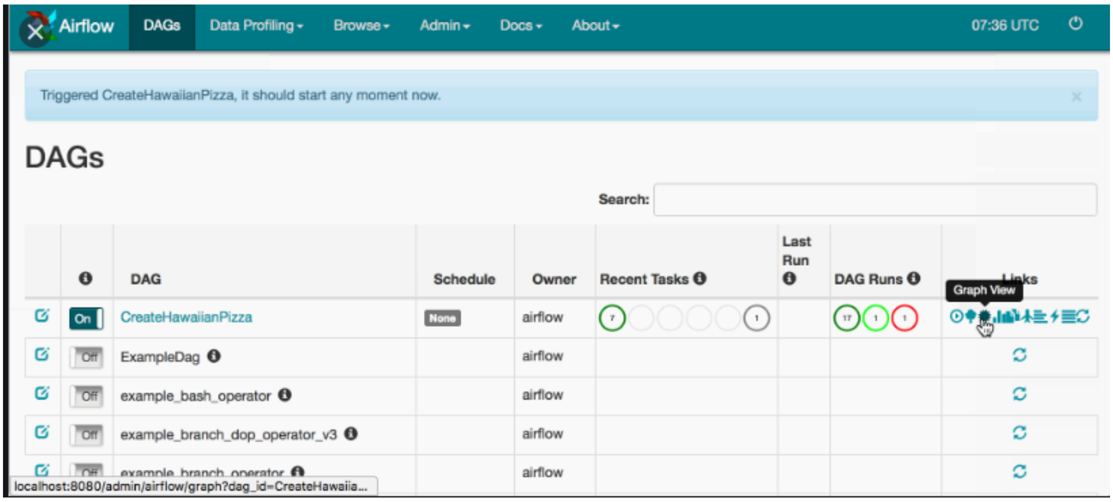
```
>airflow webserver -p 8080
```

In another prompt, run:

```
>airflow scheduler
```

Now, ope a browser tab and type:

```
localhost:8080
```



Triggered CreateHawaiianPizza, it should start any moment now.

DAGs

Search:

	ℹ	DAG	Schedule	Owner	Recent Tasks ℹ	Last Run ℹ	DAG Runs ℹ	Links
	On	CreateHawaiianPizza	None	airflow	7 ○ ○ ○ ○ 1		17 1 1	
	Off	ExampleDag ℹ		airflow				
	Off	example_bash_operator ℹ		airflow				
	Off	example_branch_dop_operator_v3 ℹ		airflow				
	Off	example_branch_operator ℹ		airflow				

localhost:8080/admin/airflow/graph?dag_id=CreateHawaiia...