

Enable Celery in your system

1. Download and install python

I'll recommend you to download the latest version of the python and install the same

Download from <https://www.python.org>

2. Install Celery

Celery can be installed using pip, using the following command to install the same

```
pip install celery
```

Confirm the installations using

```
python3 --version  
# should list Python 3.12.x or higher
```

```
pip freeze | grep celery  
#should list celery==5.3.1 or higher
```

3. Installation of the RabbitMQ - The Broker

Celery needs a broker to run which acts as a medium of communication between the python code and celery. Many brokers can be used here, but redis and rabbitmq are the ones used often.

*For this tutorial, we'll be using **RabbitMQ** and we will be using the docker image of **RabbitMQ:Management***

3.1.1

Download and Insall the Docker form <https://www.docker.com/>

Verify the docker installation using the following command

```
docker -v  
# should list Docker version 26.0.0 or Higher
```

3.2 Run Rabbitmq:Management Docker Image

*In the command prompt, please pull and use the image of **RabbitMQ:Management** using the following command*

```
docker run -it --rm --name rabbitmq -p 5672:5672 -p 15672:15672  
rabbitmq:3.12-management
```

To verify whether the RabbitMQ is working or not, please type <http://localhost:15672/> in the browser. you should see a page like this



Username: *

Password: *

Login

RabbitMQ Management

*Please provide **guest** as both username and password*

If you see a page like this, then your rabbitmq instance is working

The screenshot shows the RabbitMQ Management UI. At the top, there's a header with the RabbitMQ logo, version (3.12.4), and Erlang version (25.3.2.5). Below the header is a navigation bar with tabs: Overview, Connections, Channels, Exchanges, Queues and Streams, and Admin. The Overview page is active, showing a 'Totals' section with metrics like 'Queued messages' and 'Message rates'. Below this is a 'Nodes' section with a table of nodes. The table has columns for Name, File descriptors, Socket descriptors, Erlang processes, Memory, Disk space, Uptime, Info, and Reset stats. The first node is 'rabbit@e56dab408c8e' with 37 file descriptors, 0 socket descriptors, 403 Erlang processes, 149 MiB memory, 50 GiB disk space, and 8m 39s uptime. The bottom of the page has a footer with links to HTTP API, Documentation, Tutorials, New releases, Commercial edition, Commercial support, Google Group, Discord, Slack, Plugins, and GitHub.

4. Installation of the Flower - The Celery Management Tool

Install the flower using the following command

```
pip install flower
```

5. Running on Windows

*Celery is NOT officially supported on windows because of the **prefork pool** issue. However, for this tutorial you can use the same in windows using one of the following ways*

Single threaded instance

```
python -m celery -A tasl_one worker --loglevel=INFO --pool=solo
```

Multi threaded instance

```
python -m celery -A tasl_one worker --loglevel=INFO --pool=threads --concurrency=10
```

Using gevent

```
pip install gevent  
python -m celery -A tasl_one worker --loglevel=INFO --pool=gevent
```

All the above tools are mandatory for this tutorial, you can additionally install Prometheus & Grafana but is not necessary

6. Download and Install Prometheus

Prometheus is a time series database and we can use the same to monitor the status of celery tasks using the flower infrastructure

Please download and install the same from the following link

<https://prometheus.io/download/>

Alternatively you can also run a docker image of the prometheus

7. Install grafana or run a docker image

Grafana can be installed using the docker image as

```
docker run -d --name=grafana -p 3000:3000 grafana/grafana
```

Now everything is available in your system to make celery work