Flask – Deployment

Externally Visible Server

A Flask application on the development server is accessible only on the computer on which the development environment is set up. This is a default behavior, because in debugging mode, a user can execute arbitrary code on the computer.

If **debug** is disabled, the development server on local computer can be made available to the users on network by setting the host name as **'0.0.0.0'**.

```
app.run(host = '0.0.0.0')
```

Thereby, your operating system listens to all public IPs.

Deployment

To switch over from a development environment to a full-fledged production environment, an application needs to be deployed on a real web server. Depending upon what you have, there are different options available to deploy a Flask web application.

For small application, you can consider deploying it on any of the following hosted platforms, all of which offer free plan for small application.

- Heroku
- dotcloud
- webfaction

Flask application can be deployed on these cloud platforms. In addition, it is possible to deploy Flask app on Google cloud platform. Localtunnel service allows you to share your application on localhost without messing with DNS and firewall settings.

If you are inclined to use a dedicated web server in place of above mentioned shared platforms, following options are there to explore.

mod_wsgi

mod_wsgi is an Apache module that provides a WSGI compliant interface for hosting Python based web applications on Apache server.

Installing mod_wsgi

To install an official release direct from PyPi, you can run −

```
pip install mod_wsgi
```

To verify that the installation was successful, run the mod_wsgi-express script with the start-server command −

```
mod_wsgi-express start-server
```

This will start up Apache/mod_wsgi on port 8000. You can then verify that the installation worked by pointing your browser at −

```
http://localhost:8000/
```

Creating .wsgi file

There should be a **yourapplication.wsgi** file. This file contains the code **mod_wsgi,** which executes on startup to get the application object. For most applications, the following file should be sufficient −

```
from yourapplication import app as application
```

Make sure that **yourapplication** and all the libraries that are in use are on the python load path.

Configuring Apache

You need to tell **mod_wsgi,** the location of your application.

```
<VirtualHost *>
   ServerName example.com
   WSGIScriptAlias / C:\yourdir\yourapp.wsgi

   <Directory C:\yourdir>
      Order deny,allow
      Allow from all
   </Directory>

</VirtualHost>
```

Standalone WSGI containers

There are many popular servers written in Python that contains WSGI applications and serve HTTP.

- Gunicorn
- Tornado
- Gevent
- Twisted Web

Revision #1
Created 14 December 2022 11:30:04 by Admin
Updated 14 December 2022 11:30:27 by Admin