Bartender, Whiskey is served here.

None

None

Table of contents

1. Bartender: a wsgi server

This impliments a wsgi server that is called by a github webhook. When called it validates the request, schedules an at job, and returns a status. The at job pulls the "docs" off of github, builds the site, and moves it into place.

1.1 The Stack

```
graph LR
    B -- http(s)://127.0.0.1:8000 <--> C[nginx] -- http(s)://bartender/whisky/STYLE <--> I([Internet])
    A[flask] -- wsgi <--> B[gunicorn];
```

The implimentation

```
graph LR
   A[bartender app] --> C[AT]
   A -- 200 ok / 400 bad id / 404  --> N[NGinX]
   N <-- bartender.digithink.com --> I([Internet])
   G[(github)] -- pull --> D
   C --> D[makesite.sh]
   D -- mkdocks build --> O[(site)] -->N
   N -- whiskey/STYLE --> A
```

1.2 The source code and the results

- https://github.com/suspect-devices/bartender The source code for the bartender service serves the it's documentation as well as several other sites.
- https://www.digithink.com is the one of the target websites.
- https://www.3dangst.com is another one of the target website.
- https://bartender.digithink.com/ contains this and the other documents for the service.

1.2.1 Installing the service

```
apt install nginx
apt install certbot python3-certbot-nginx
apt install python3-flask
apt install python3-gunicorn
apt install at

echo www-data |tee /etc/at.allow
nano /etc/systemd/system/whiskey.service
[Unit]
Description=Gunicorn instance to serve whiskey
After=network.target

[Service]
User=www-data
Group=www-data
WorkingDirectory=/var/www/bartender/repo/whiskey
#Environment="PATH=/home/sammy/myproject/myprojectenv/bin"
#     --bind unix:/run/whiskey.sock  \

ExecStart=/usr/bin/gunicorn \
     --workers 3 \
     --bind 127.0.0.1:5000 \
     --reload \
     --access-logfile /var/www/bartender/repo/whiskey/logs/gunicorn_access.log \
     --error-logfile /var/www/bartender/repo/whiskey/logs/gunicorn_error.log \
     -m 007 wsgi:app
ExecReload=/bin/kill -s HUP $MAINPID
ExecStop=/bin/kill -s TERM $MAINPID

[Install]
WantedBy=multi-user.target
^X
systemctl enable whiskey
systemctl start whiskey
```

1.2.2 nginx.conf

```
# lots of hard coded foo here
server {
    server_name bartender.digithink.com;

    listen 198.202.31.232:443 ssl;
    server_name bartender.digithink.com;
    ssl_certificate /etc/letsencrypt/live/bartender.digithink.com/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/bartender.digithink.com/privkey.pem; # managed by Certbot

    root /var/www/bartender/site;
    index index.html;
    location /whiskey {
        include proxy_params;
        proxy_pass http://bartender/whiskey;
    }

    error_page 404 /404.html;
    location  /404.html {
          internal;
    #     root  /var/www/digithink/site;
    }
    location /lacuenta { # FIX THIS
        root /var/www/bartender/whiskey/logs;
    }
}

server {
    if ($host = bartender.digithink.com) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    listen 198.202.31.232:80;
    server_name bartender.digithink.com;
    return 404; # managed by Certbot
}
```

1.2.3 the WSGI app

The actual app will grow into something with better feedback more general use (ie to make different static web sites)

drink.py

Minimum Viable Product

```
from flask import Flask
from markupsafe import escape
import subprocess

app = Flask(__name__)

@app.route("/whiskey/<style>",methods = ['POST'])
def whiskey(style):
    # break this out by style.
    subprocess.call(['at', 'now', '-f', '/var/www/digithink/repo/makesite.sh'])
    return f"One Whiskey, {escape(style)}!"
```

The actual drink.py is slightly more developed.

wsgi.py, Turning the above into a WSGI app

```
from drink import app

if __name__ == "__main__":
    app.run()
```

1.3 Converting the git content to a static html site.

1.3.1 mkdocs plus the extensions

Ubuntu (... ok, debian really...) fracked up the packaging for mkdocs and mkdocs-material. I wound up removing the packages and pip3 installing most of it with --break-system-packages.

```
apt remove mkdocs*
apt remove markdown
apt remove python3-markdown
apt install python3-regex
apt install libvips-dev
apt install python3-pip
pip3 install mkdocs-material --break-system-packages
pip3 install mkdocs-with-pdf -U git+https://github.com/domWalters/mkdocs-to-pdf.git@release-v0.9.4 --break-system-packages
```

pull dependencies based on the current mkdocs install and mkdocs.yml

```
cd /var/www/digithink/&& git pull
mkdocs-get-deps > requirements.txt
pip3 install $(mkdocs-get-deps) --break-system-packages
mkdocs build && chown -R www-data:www-data site/
```

1.4 linkdump

- https://github.com/codingforentrepreneurs/Pi-Awesome/blob/main/how-tos/Create%20a%20Minimal%20Web%20Application%20with%20Nginx%2C%20Python%2C%20Flask%20%26%20Raspberry%20Pi.md
- https://www.stackovercloud.com/2020/05/27/how-to-serve-flask-applications-with-uwsgi-and-nginx-on-ubuntu-20-04/
- https://www.digitalocean.com/community/tutorials/how-to-set-up-uwsgi-and-nginx-to-serve-python-apps-on-ubuntu-14-04
- https://stackoverflow.com/questions/10748108/nginx-uwsgi-unavailable-modifier-requested-0#11055729
- https://github.com/mkdocs/get-deps