Mastodon

Install Kutt.it

- Kutt.it - a Free, Self Hosted, Open Source URL Shortener

Kutt.it - a Free, Self Hosted, Open Source URL Shortener

https://www.youtube.com/embed/pm_sL4rfxu4

Kutt.it is a great little application. It was suggested by a subscriber, and I really have like it. It's got a simple, clean interface, and does exactly what you expect from a link shortening application.

Installation is pretty straight forward, and like most things I cover we'll use Docker and Docker-Compose

What you'll need

- A server or machine with Docker and Docker-compose installed.
- A Domain Name you own, and can set an A-record for with the public IP of your server.
- (Suggested) NGinX Proxy Manager to proxy traffic and get LetsEncrypt Certificates for the application.

Installation of Docker and Docker-Compose

If you are using Ubuntu 18.04 or 20.04, then you can use the instructions below to make a simple script that will install Docker for you. IF you prefer, you can simply copy the lines from the script (except the first line #!/bin/bash and any line with a # in front of it) and run then one by one.

If you want to create the script first you want to make a file called "install_docker.sh"

```
nano install_docker.sh
```

Once in the file, you want to add the following lines:

```
#!/bin/bash
sudo apt update
sudo apt install apt-transport-https ca-certificates curl software-properties-common -y
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
# for Ubuntu 20.04 use this line and put a # in front of the line below for 18.04
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable"

#for Ubuntu 18.04 use this line and put a # in front of the line above for 20.04
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable"

sudo apt update
apt-cache policy docker-ce
sudo apt install docker-ce -y

# add current user to docker group so sudo isn't needed
sudo usermod -aG docker ${USER}
```

Once this runs, you need to log out and back in (i prefer to reboot completely), so that the changes will allow your user to run the `docker` command without sudo.

Install NGinX Proxy Manager

NGinX Proxy Manager (NPM) needs to have port 80 and port 443 available on the machine you install it on. If you already have a container using 80 and 443 on this machine, you'll want to stop that container, and then install and start NPM. Next, make any necessary adjustments to the other container to use other ports.

To install NPM you need to install docker and docker-compose, and create a new folder on the server you want to run it in. Next, you'll create two files inside that folder:

- config.json
- docker-compose.yml

Inside the config.json file, you'll put the following:

```
{
  "database": {
    "engine": "mysql",
    "host": "db",
    "name": "npm",
    "user": "<your desired username>",
    "password": "<a strong password>",
    "port": 3306
  }
}
```

And inside the docker-compose.yml file you'll put:

```
version: '3'
services:
  app:
    image: 'jc21/nginx-proxy-manager:latest'
    ports:
      - '80:80'
      - '81:81'
      - '443:443'
    volumes:
      - ./config.json:/app/config/production.json
      - ./data:/data
      - ./letsencrypt:/etc/letsencrypt
  db:
    image: 'jc21/mariadb-aria:10.4'
    environment:
      MYSQL_ROOT_PASSWORD: 'npm'
      MYSQL_DATABASE: 'npm'
      MYSQL_USER: '<username from config.json>'
      MYSQL_PASSWORD: '<strong password from config.json>'
    volumes:
      - ./data/mysql:/var/lib/mysql
```

Make sure to replace the items with < and > around it in each file, and that the username and passwords in each file match.

Now run the command:

```
docker-compose up -d
```

Give it a minute to pull down everything, and get started, and then in your browser go to the IP address of your server. You should get a Congratulations screen.

if you go tot he IP address at port 81 (http://192.168.1.x:81), you'll be prompted to login to NPM.

Default credentials are:

username: admin@example.com
passwrod: changeme

Make sure to update the email and password, from the default values, then log out, and back in usign the new values you entered.

Now, you're ready to start proxying traffic.

Install Kutt.it

To install Kutt.it, you'll need to pull down the repo from Github.

```
git clone https://github.com/thedevs-network/kutt.git
```

Next, move into the folder it creates.

```
cd kutt
```

Now, you can look at the contents with the command

```
ls -al
```

You'll see a few files that start with a dot "." . These are hidden files in the linux/unix world.

Next, we need to copy the .docker.env file to a file called ".env".

```
cp .docker.env .env
```

Now, we need to edit the .env file contents we just copied.

```
nano .env
```

Inside this file we need to edit the following fields:

SITE_NAME (the short name of your site)
DEFAULT_DOMAIN (you need a domain for this to work, for instance mine is osia.me)
LINK_LENGTH (optional)
DISALLOW_LOGIN (optional)
DISALLOW_ANONYMOOUS_LINKS (optional)
USER_LIMIT_PER_DAY (optional)
NON_USER_COOLDOWN (optional)
JWT_SECRET (make this a long strong secret - not something you have to remember)
ADMIN_EMAILS (emails that are comma separated for admins fo the software)

Email Section: This is necessary if you want registration to work. If you don't intend to use Registration, just don't mess with the Email setting section.

MAIL_HOST
MAIL_PORT (needs to be 465 and SSL as Kutt does not yet support StartTLS on 587)
MAIL_SECURE=true
MAIL_USER
MAIL_FROM
MAIL_PASSWORD

REPORT_EMAIL (optional)

Save the file with CTRL+O, then Enter / Return, and exit with CTRL+X.

Next, we need to open the file called "docker-compose.yml".

```
nano docker-compose.yml
```

In this file you need to potentially edit the following values:

Under the section labeled "kutt", you may need to change the port on the left side fo the colon. By default it's set as "3000:3000". If you have port 3000 already in use on your host, you should change the left side of the colon to a free port. I changed mine to be 3030, so it now looks like this:

```
"3030:3000"
```

Next, you need to change the environment valued under the section labeled "kutt" and "postgres", and ensure these values match.

DB_USER = POSTGRES_USER
DB_PASSWORD = POSTGRES_PASSWORD
DB_NAME = POSTGRES_DB

Set the values to what you want, just ensure they are equal in those two sections.

Once you've made these adjustments, save with CTRL+O, then Enter / Return to save, and CTRL+X to exit.

Run the Kutt Server

We now just need to run Kutt.it with the command

```
docker-compose up -d
```

Give it time to pulldown the necessary images, then give it a minute after you see 'done' in the terminal. You'll likely not be able to reach it via the IP address and port, but instead need to set it up to be accessed via the domain name you procured.

For this, I'll be using NGinX Proxy Manager.

Setup Kutt for Access via Domain Name in NPM

Go to NGinX Proxy Manager (server IP and port 81 - for example my server IP is 192.168.7.125, so I go to http://192.168.7.125:81). Login, hopefully you got all this setup from the information above. If not, go back and get it setup, then continue.

Add a new Proxy Host, and in the Details tab of the pop-up enter your registered domain name. I registered "osia.me", so that's what I'll enter, then press tab to make sure it's accepted.

Next, in the IP address field, we want to enter the docker0 IP, so we get that in the terminal ssh'ed into the machine, by using the command:

```
ip addr show docker0
```

Usually this IP is 172.17.0.1, so I put that in the IP field, and I enter the port that I set in the docker-compose.yml file in the port field. In my case I used 3030.

Now I enable Websocket Support (though not sure it's needed), and save.

I test by clicking the URL in the list on the NPM page. It should open a new tab, and take you to your running Kutt.it server.

Now Setup SSL

Once you can reach the page through http, we want to set it up to run with https (SSL encryption). We'll go back to NGinX Proxy Manager and edit our entry.

On the SSL tab of the pop-up, select "Request a New Certificate" from the drop down, then enable the Force SSL option. Enter your email address in the field, and Accept the Terms of Service for LetsEncrypt.

Click Save again, and if you've got everything setup properly, a new SSL certificate will be added for your page.

Now, if you click the URL in the list on NPM, you'll be taken to a new tab that will be through https.

Voila! You've done it. If you hit issues along the way, make sure to go back and check my video for any clues.

Support my Channel and Content

Support my Channel and ongoing efforts through Patreon:
https://patreon.com/awesomeopensource