Sammy welcomes you to your Droplet!

Things to do with this script

This message is coming to you via a simple NodeJS application that's live on your Droplet! This droplet is all set up with NodeJS, PM2 for process management, and nginx.

Run all pm2 commands using the nodejs user or a second instance of pm2 will start. The login and password are stored in the `NODE_USER*` values you see when you call `cat /root/.digitalocean_passwords` while logged in over SSH.

This app is running at port 3000, and is being served to you by nginx, which has mapped port 3000 to be served as the root URI over HTTP (port 80) -- a technique known as a "reverse proxy." We'll be teaching you how to use this technique right here on this page. If you want to kick the tires right now, try some of these things:

- SSH into your Droplet and modify this script at `/var/www/html/hello.js` and see the results live by calling `pm2 restart hello`
- Run `pm2 list` to see code scheduled to start at boot time
- Run `pm2 delete hello` to stop running this script and `pm2 save` to stop it from running on Droplet boot

Get your code on here

- SSH into your Droplet, and `git clone` your NodeJS code onto the droplet, anywhere you like
  - Note: If you're not using a source control, you can directly upload the files to your droplet using SFTP.
- `cd` into the directory where your NodeJS code lives, and install any dependencies. For example, if you have a `package.json` file, run `npm install`.
- Launch your app by calling `pm2 start <your-file>`, then map the port your app runs on to an HTTP URL by running `nano /etc/nginx/sites-available/default` and adding another `location`. Use the existing entry for the port 3000 "hello" app as a basis.
- Call `sudo systemctl restart nginx` to enable your new nginx config.
- Call `pm2 save` to schedule your code to run at launch.
- Repeat these steps for any other NodeJS apps that need to run concurrently -- schedule them to run at boot time on whatever internal port you like using PM2, then map that port to an HTTP/HTTPS

URL in the nginx config. Build out the URL directory structure you need by mapping applications to URL paths; that's the reverse proxy method in a nutshell!

Get production-ready

There's a lot you'll want to do to make sure you're production-ready.

- Set up a non-root user for day-to-day use
- Review your firewall settings by calling `sudo ufw status`, and make any changes you need. By default, only SSH/SFTP (port 22), HTTP (port 80), and HTTPS (port 443) are open. You can also disable this firewall by calling `sudo ufw disable` and use a DigitalOcean cloud firewall instead, if you like (they're free).
- Register a custom domain
- Have data needs? You can mount a volume (up to 16TB) to this server to expand the filesyem, provision a database cluster (that runs MySQL, Redis, or PostgreSQL), or use a Space, which is an S3-compatible bucket for storing objects.