Quickstart

Get going fast! Intended for folks familiar with setting up DevOps environments. These instructions were tested using Ubuntu 20.04. Ubuntu 18x is no longer supported because the versions of Python3 available on it are not current.

PostgreSQL Installation

Gain access to an Ubuntu 18.04 or later environment and install PostgreSQL. Ubuntu 20.04 is recommended because its long-term support (LTS) window is longer.

```
sudo apt update
sudo apt upgrade
sudo apt install software-properties-common
sudo apt install python3-dev
sudo apt install postgresql postgresql-contrib postgresql-client
sudo apt install build-essential
```

Create a PostgreSQL database for Augur to use:

```
sudo su -
su - postgres
psql
```

Then, once you’ve connected to your PostgreSQL instance:

```
postgres=# CREATE DATABASE augur;
postgres=# CREATE USER augur WITH ENCRYPTED PASSWORD 'password';
postgres=# GRANT ALL PRIVILEGES ON DATABASE augur TO augur;
```

Git Configuration

Configure Git: These instructions assume the potential of large repositories that occasionally perform significant refactoring within a small number of commits. Our experience is that nearly all organizations have at least one project that meets these criteria.

```
git config --global diff.renames true
git config --global diff.renameLimit 200000
git config --global credential.helper cache
git config --global credential.helper 'cache --timeout=9999999999999'
```

For each platform, perform a command-line login in order to cache Git credentials for the Linux user who operates Augur. This step is required in order to prevent the Facade Commit Counting Diesel from stalling on a cmd line prompt when repositories move or disappear.

Install Go

Two of Augur's workers use the Go programming language, which needs to be installed on your computer. Snap is the easiest way to install Go. If Snap does not work for you, see instructions here:
https://www.digitalocean.com/community/tutorials/how-to-install-go-on-ubuntu-20-04

```
sudo apt update
sudo apt install snapd
sudo snap install go --classic
```

Install NPM

```
sudo curl https://raw.githubusercontent.com/creationix/nvm/master/install.sh | bash
source ~/.profile
nvm --version
nvm install node
node --version
npm --version
```

Export Digital Envelope

A digital envelope allows users to encrypt data with the speed of secret key encryption.

```
export NODE_OPTIONS=--openssl-legacy-provider
```

Python Virtual Environment Configuration

Set up a Python virtual environment (Python 3.8 and above are now required. Python 3.9 and Python 3.10 work as well, though we have tested Python 3.9 on more platforms.) Clone and install Augur as a regular user.

```
cd ~
git clone https://github.com/chaoss/augur.git
cd augur/
sudo apt install make
sudo apt-get install python3-venv
python3 -m venv $HOME/.virtualenvs/augur_env
source $HOME/.virtualenvs/augur_env/bin/activate
sudo apt install python-pip-whl
sudo apt install python3-pip
sudo apt install pythonpy
python -m pip install --upgrade pip
make install-dev
nohup augur backend start >logs/base.log 2>logs/base.err &
```

Docker Quick Start

Linux is currently the only supported platform for the script. Docker is slightly different on macOS. Additionally, the script uses a network alias for local connections which is done differently for macOS. The script will set up the alias for macOS correctly but it is untested for macOS and can be unpredictable.

Credentials

Before you get started with Docker, you'll need to set up a PostgreSQL instance either locally or using a remote host. Alternatively, you can also set up the database within a docker container either manually or through the script but this is not recommended.

```
cd /etc/postgresql/12/main/
sudo vi postgresql.conf
>>> listen_addresses = '*'
sudo vi pg_hba.conf
>>> host all all 0.0.0.0/0 md5
sudo /etc/init.d/postgresql restart
```

Install Docker

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay.

```
sudo apt-get update

sudo apt-get install \
apt-transport-https \
ca-certificates \
curl \
gnupg-agent \
software-properties-common

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
sudo apt-key fingerprint 0EBFCD88

sudo add-apt-repository \
"deb [arch=amd64] https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) \
stable"
```

```
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io
sudo usermod -aG docker $USER
sudo systemctl enable docker
docker --version

sudo curl -L \
"https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" \
-o /usr/local/bin/docker-compose

sudo chmod +x /usr/local/bin/docker-compose
docker-compose --version
```

Docker Compose Deployment

Compose is a tool for defining and running multi-container Docker applications. With Compose, you use a YAML file to configure your application's services. Then, with a single command, you create and start all the services from your configuration.

```
sudo apt install net-tools
sudo ./docker-setup.sh
docker-compose -f frontend-compose.yml up -d
docker ps
docker logs augur_database_1
docker logs augur_backend_1
docker logs augur_frontend_1
```

Using Compose is basically a three-step process:

- Define your app's environment with a Dockerfile so it can be reproduced anywhere.
- Define the services that make up your app in docker-compose.yml so they can be run.
- Run docker-compose up and the Docker compose command starts and run your entire app. You can alternatively run docker-compose up using the docker-compose binary.

Get & Generate Augur API Key

```
docker exec -it augur_backend_1 augur db get-api-key
docker exec -it augur_backend_1 augur db generate-api-key
```

Check Database

```
docker exec -it augur_database_1 psql -U augur -d augur
augur=# \l
augur=# \c augur
augur=# \dt
augur=# \q
```