1. installing Docker

## 1.0 Playing with Busybox

$ docker pull busybox: download busybox image, fetches the busybox [**image**](https://hub.docker.com/_/busybox/) from the [**Docker registry**](https://hub.docker.com/explore/) and saves it to our system

$ docker run busybox:

Based on the image create and run busybox container

Sh: interactive mode, when the command is running, we can enter commands

Rm: remove the container

### 1.2 Terminology

### 2.0 Webapps with Docker

### 2.1 Static Sites

### $ docker port static-site; shows the default port

### After running the container using the default port 32769, You can open [http://localhost:32769](http://localhost:32769/) in your browser and see the static website.

$ docker run -p 8888:80 prakhar1989/static-site

:To specify the port: 8888, then use localhost:8888

### 2.2 Docker Images

Concepts of images

### 2.3 Our First Image: create images by ourselves

$ git clone https://github.com/prakhar1989/docker-curriculum:

Clone repository locally, or git clone source code of website, downloaded from other website

### 2.4 Dockerfile: used to create image

FROM python:3-onbuild: base image

EXPOSE 5000: for others to connect to the container

CMD ["python", "./app.py"]: use python, run the app called app.py

Catnip: name of the website

$ docker build -t prakhar1989/catnip .

: create a container. Here “.” Means local folder running dockfile

$ docker run -p 8888:5000 prakhar1989/catnip

: run

### 2.5 Docker on AWS

$ docker push prakhar1989/catnip

: push to a public repository, so others can use it from other computers

##### **Beanstalk**

Follow the steps, finally, container is running on Beanstalk

## 3.0 Multi-container Environments

App name: SF Food Trucks

The app's backend is written in Python (Flask) and for search it uses [Elasticsearch](https://www.elastic.co/products/elasticsearch).

$ docker search elasticsearch:

Search the image

$ docker run -dp 9200:9200 elasticsearch: run it

$ docker build -t prakhar1989/foodtrucks-web .:

Build backend

Fail, because there is no network between the containers

### 3.2 Docker Network

Bash: enter this container to execute this command.

es = Elasticsearch(host='es'): es is the IP of the ES container

In the example, the foodtrucks-web try to ping IP of the ES container by doing:

root@35180ccc206a:/opt/flask-app# curl 172.17.0.2:9200

$ docker network create foodtrucks

: create our own network

$ docker stop e931ab24dedc

:Then stop the originally running ES container

$ docker run -dp 9200:9200 --net foodtrucks --name es elasticsearch

: create and run ES container and connect it with the created network

$ docker run -it --rm --net foodtrucks prakhar1989/foodtrucks-web bash

: create and run backend container and connect it with the created network

cat /etc/hosts

: check all network connection information

root@53af252b771a:/opt/flask-app# exit

: stop it

Then, run it again:

$ docker run -d --net foodtrucks -p 5000:5000 --name foodtrucks-web prakhar1989/foodtrucks-web

2a1b77e066e646686f669bab4759ec1611db359362a031667cacbe45c3ddb413

$ git clone https://github.com/prakhar1989/FoodTrucks

$ cd FoodTrucks

$ ./setup-docker.sh

--does all the above work using one command

### 3.3 Docker Compose

Install docker compose

$ docker-compose up -d: run it

$ docker-compose stop: stop

### 3.4 AWS Elastic Container Service (ECS)

Enter AWS key-pair

$ ecs-cli up --keypair ecs --capability-iam --size 2 --instance-type t2.micro

--created a VM cluster

Ecs-cli: command to connect to ECS

Create a cluster, push to the public repository, let Ec2 to run