See this to setup docker container and mongodb inside it:
http://www.cs.sjsu.edu/~kim/nosql/contents/handouts/MongoDBSingleNodeSetup.pdf

Create a ubuntu container and go inside the container:

$ docker run -it --name mynode ubuntu:latest

This command does the following 3 actions:

1. Downloads the Docker image for the latest version of Ubuntu Linux OS from Docker Hub (https://hub.docker.com) to your laptop, if it's the first time.
2. Using the downloaded image, it creates a Docker container called "mynode".
3. Lets you go inside the "mynode" container.

Inside the container, install necessary packages:

```
root@mynode:/# apt-get update
root@mynode:/# apt-get upgrade -y
root@mynode:/# apt-get install -y vim wamerican iputils-ping iproute2 curl wget sudo gnupg git
root@mynode:/# which git
```

Download the mongod service script (still inside the container):

```
root@mynode:/# cd /tmp
root@mynode:/tmp# git clone https://github.com/kimsjsu/service.git
root@mynode:/tmp# cd service
root@mynode:/tmp/service# ls
service-script-mongod
root@mynode:/tmp/service# cp -p service-script-mongod /etc/init.d/mongod
root@mynode:/tmp/service# chmod 755 /etc/init.d/mongod
root@mynode:/tmp/service# update-rc.d mongod defaults
root@mynode:/tmp/service# service --status-all
 [ ? ]  hwclock.sh
```

https://docs.mongodb.com/manual/tutorial/install-mongodb-on-ubuntu/

1. wget -qO - https://www.mongodb.org/static/pgp/server-4.2.asc | sudo apt-key add -
2. echo "deb [arch=amd64,arm64] https://repo.mongodb.org/apt/ubuntu bionic/mongodb-org/4.2 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.2.list
3. sudo apt-get update
4. sudo apt-get install -y mongodb-org
    a. echo "mongodb-org hold" | sudo dpkg --set-selections
    b. echo "mongodb-org-server hold" | sudo dpkg --set-selections
    c. echo "mongodb-org-shell hold" | sudo dpkg --set-selections
    d. echo "mongodb-org-mongos hold" | sudo dpkg --set-selections
    e. echo "mongodb-org-tools hold" | sudo dpkg --set-selections

In docker toolbox, at each window:

- docker run -it `-d -p 27119:27021` --name myMongos ubuntu:latest
- docker run -it --name mConfigs ubuntu:latest
- docker run -it --name myShard0 ubuntu:latest
- docker run -it --name myShard1 ubuntu:latest
- docker run -it --name myShard2 ubuntu:latest

Use links/steps above to set up each docker container.

Use this to check container IP:

docker inspect containerName

```
myMongos 172.17.0.2
myConfigs 172.17.0.3
myShard0 172.17.0.4
myShard1 172.17.0.5
myShard2 172.17.0.6
```

Follow this for create replica set and sharding
https://docs.google.com/document/d/1gTnjmT71Gf7UYdm91xD5QUM_gnQewmsw8XMnpLKurE4/edit

After the db is set up, in the program:

```
import pymongo

client = pymongo.MongoClient("192.168.99.100", 27119)
print(client.list_database_names())
```

Where docker toolBox ip is 192.168.99.100, and 27119 is the exposed port in the myMongos container with port 27021

Mongos instance/node (host/ip 172.17.0.2):

- port#27017 mongos

configServer instance/node (host/ip 172.17.0.3):

- port#27020 config server PRIMARY
- port#27021 config server SECONDARY
- port#27022 config server SECONDARY

Shard0 instance/node (host/ip 172.17.0.4):

- port#27023 shard0 PRIMARY
- port#27024 shard0 SECONDARY
- port#27025 shard0 SECONDARY

Shard1 instance/node (host/ip 172.17.0.5):

- port#27026 shard1 PRIMARY
- port#27027 shard1 SECONDARY

- port#27028 shard1 SECONDARY

Shard2 instance/node (host/ip 172.17.0.6):

- port#27029 shard1 PRIMARY
- port#27030 shard1 SECONDARY
- port#27031 shard1 SECONDARY

Create directory

- Config
  - sudo **mkdir -p /db/config1/data /db/config2/data /db/config3/data**
- Shard1
  - sudo **mkdir -p /db/shard0/data1 /db/shard0/data2 /db/shard0/data3**
- Shard2
  - sudo **mkdir -p /db/shard1/data1 /db/shard1/data2 /db/shard1/data3**
- Shard3
  - sudo **mkdir -p /db/shard2/data1 /db/shard2/data2 /db/shard2/data3**

Config

- sudo mongod --bind_ip 172.17.0.3 --port 27020 --dbpath /db/config1/data --configsvr --replSet configset
- sudo mongod --bind_ip 172.17.0.3 --port 27021 --dbpath /db/config2/data --configsvr --replSet configset
- sudo mongod --bind_ip 172.17.0.3 --port 27022 --dbpath /db/config3/data --configsvr --replSet configset
- mongo --host 172.17.0.3--port 27020
  - rs.initiate()
  - rs.status()
  - rs.add("172.17.0.3:27021")
  - rs.add("172.17.0.3:27022")
  - rs.status()

Mongos

- mongos --configdb configset/172.17.0.3:27020,172.17.0.3:27021,172.17.0.3:27022 --bind_ip 172.17.0.2 --port 27017
- mongo --host 172.17.0.2 --port 27017
- sh.status()

Shard0

- sudo mongod --bind_ip 172.17.0.4 --port 27023 --dbpath /db/**shard0**/data1 --shardsvr --replSet rs0

- sudo mongod --bind_ip 172.17.0.4 --port 27024 --dbpath /db/**shard0**/data2 --shardsvr --replSet rs0
- sudo mongod --bind_ip 172.17.0.4 --port 27025 --dbpath /db/**shard0**/data3 --shardsvr --replSet rs0
- mongo --host 172.17.0.4 --port 27023
- rs.initiate()
- rs.add("172.17.0.4:27024")
- rs.add("172.17.0.4:27025")

Shard1

- sudo mongod --bind_ip 172.17.0.5 --port 27026 --dbpath /db/**shard1**/data1 --shardsvr --replSet rs1
- sudo mongod --bind_ip 172.17.0.5 --port 27027 --dbpath /db/**shard1**/data2 --shardsvr --replSet rs1
- sudo mongod --bind_ip 172.17.0.5 --port 27028 --dbpath /db/**shard1**/data3 --shardsvr --replSet rs1
- mongo --host 172.17.0.5 --port 27026
- rs.initiate()
- rs.add("172.17.0.5:27027")
- rs.add("172.17.0.5:27028")

Shard2

- sudo mongod --bind_ip 172.17.0.6 --port 27029 --dbpath /db/**shard2**/data1 --shardsvr --replSet rs2
- sudo mongod --bind_ip 172.17.0.6 --port 27030 --dbpath /db/**shard2**/data2 --shardsvr --replSet rs2
- sudo mongod --bind_ip 172.17.0.6 --port 27031 --dbpath /db/**shard2**/data3 --shardsvr --replSet rs2
- mongo --host 172.17.0.6 --port 27029
- rs.initiate()
- rs.add("172.17.0.6:27030")
- rs.add("172.17.0.6:27031")

In Mongos instance

- sh.addShard("rs0/172.17.0.4:27023,172.17.0.4:27024,172.17.0.4:27025")

- sh.addShard("rs1/172.17.0.5:27026,172.17.0.5:27027,172.17.0.5:27028")
- sh.addShard("rs2/172.17.0.6:27029,172.17.0.6:27030,172.17.0.6:27031")

    - sh.enableSharding("yelpdb")
    - sh.shardCollection("yelpdb.yelpcollection", {user_id:1})

Useful commands:
Import dataset into mongo
mongoimport --db=dbName --collection=collectionName --file=fileName

Import file from local machine to docker container

```
docker cp [OPTIONS] SRC_PATH+filename    CONTAINER_NAME:DESTINATION_PATH+newfilename
```

```
Data wrangle after data is imported into database
```

- the elite and friends fields are not in a good json file format (they contain a string of items separated by comma without any bracket). This means that we have to manually update the fields, friends and elite, into an array after the data is inside the database by using the following command:
    - To change friends field from string to array of strings:
        - db.yelpcollection.find().forEach(function (blog) { if (blog.friends) { blog.friends=blog.friends.split(','); db.yelpcollection.save(blog); } });
    - To change elite field:
        - db.yelpcollection.find().forEach(function (blog) { if (blog.elite) { blog.elite=blog.elite.split(','); db.yelpcollection.save(blog); } });
        - db.yelpcollection.find({elite: ""}).forEach(function (blog) { blog.elite=[]; db.yelpcollection.save(blog); });