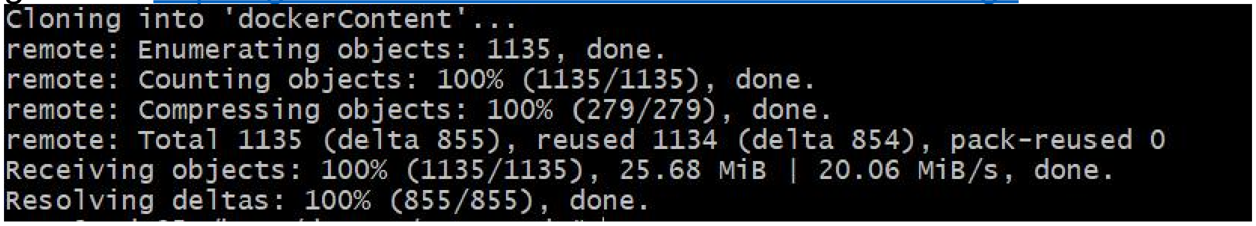
Download your company's website files from the given link :

mkdir CaseStudy

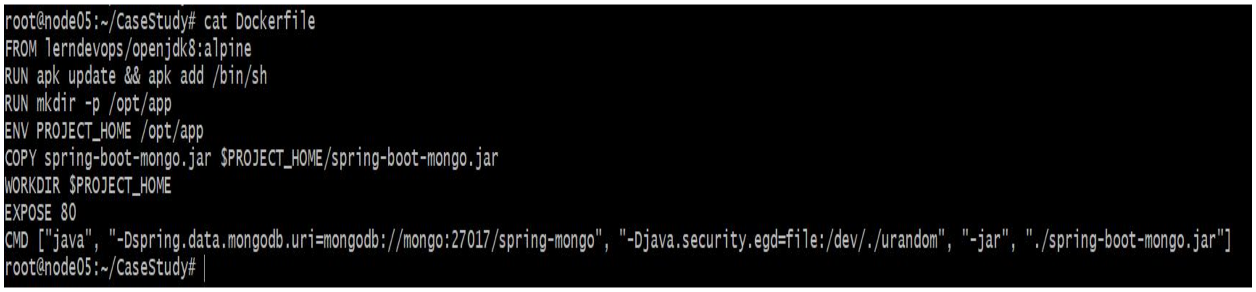
cd CaseStudy/

git clone https://github.com/edurekacontent/dockerContent.git

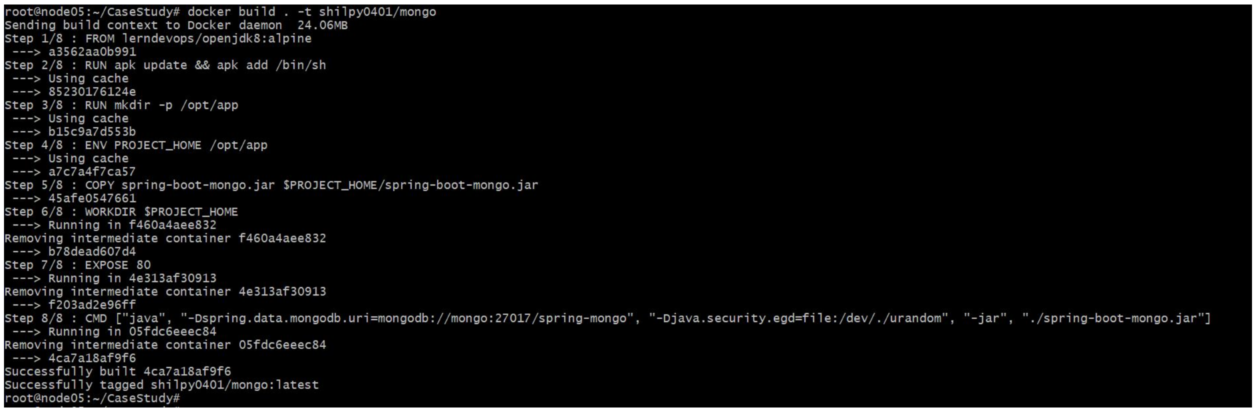


```
Cloning into 'dockerContent'...
remote: Enumerating objects: 1135, done.
remote: Counting objects: 100% (1135/1135), done.
remote: Compressing objects: 100% (279/279), done.
remote: Total 1135 (delta 855), reused 1134 (delta 854), pack-reused 0
Receiving objects: 100% (1135/1135), 25.68 MiB | 20.06 MiB/s, done.
Resolving deltas: 100% (855/855), done.
```

Write a docker file that will make your company's website work out of the box with a web server (Tip You can use httpd / apache image and build on top of it)

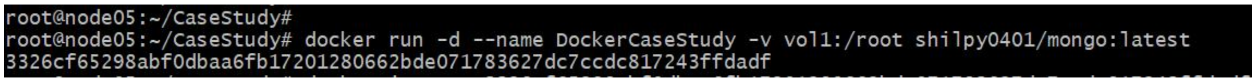


```
root@node05:~/CaseStudy# cat Dockerfile
FROM lerndevops/openjdk8:alpine
RUN apk update && apk add /bin/sh
RUN mkdir -p /opt/app
ENV PROJECT_HOME /opt/app
COPY spring-boot-mongo.jar $PROJECT_HOME/spring-boot-mongo.jar
WORKDIR $PROJECT_HOME
EXPOSE 80
CMD ["java", "-Dspring.data.mongodb.uri=mongodb://mongo:27017/spring-mongo", "-Djava.security.egd=file:/dev/./urandom", "-jar", "./spring-boot-mongo.jar"]
root@node05:~/CaseStudy# |
```



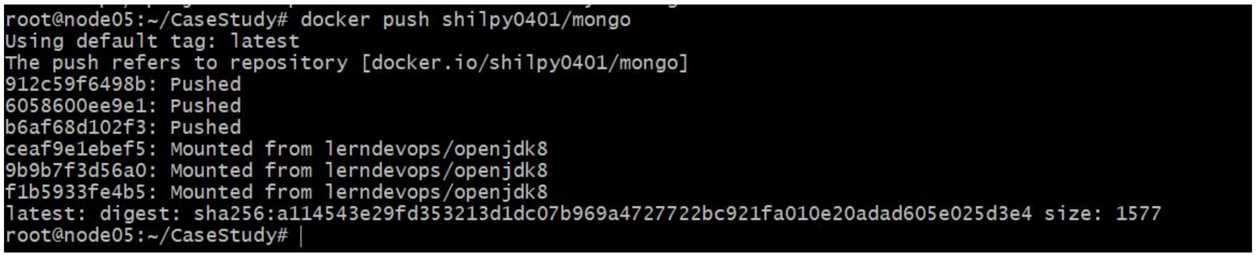
```
root@node05:~/CaseStudy# docker build . -t shilpy0401/mongo
Sending build context to Docker daemon  24.06MB
Step 1/8 : FROM lerndevops/openjdk8:alpine
 ---> a3562aa0b991
Step 2/8 : RUN apk update && apk add /bin/sh
 ---> Using cache
 ---> 85230176124e
Step 3/8 : RUN mkdir -p /opt/app
 ---> Using cache
 ---> b15c9a7d553b
Step 4/8 : ENV PROJECT_HOME /opt/app
 ---> Using cache
 ---> a7c7a4f7ca57
Step 5/8 : COPY spring-boot-mongo.jar $PROJECT_HOME/spring-boot-mongo.jar
 ---> 45afe0547661
Step 6/8 : WORKDIR $PROJECT_HOME
 ---> Running in f460a4aee832
Removing intermediate container f460a4aee832
 ---> b78dead607d4
Step 7/8 : EXPOSE 80
 ---> Running in 4e313af30913
Removing intermediate container 4e313af30913
 ---> f203ad2e96ff
Step 8/8 : CMD ["java", "-Dspring.data.mongodb.uri=mongodb://mongo:27017/spring-mongo", "-Djava.security.egd=file:/dev/./urandom", "-jar", "./spring-boot-mongo.jar"]
 ---> Running in 05fdc6eeec84
Removing intermediate container 05fdc6eeec84
 ---> 4ca7a18af9f6
Successfully built 4ca7a18af9f6
Successfully tagged shilpy0401/mongo:latest
root@node05:~/CaseStudy#
```

Make sure that you use volumes to store the actual data of the website outside of the container

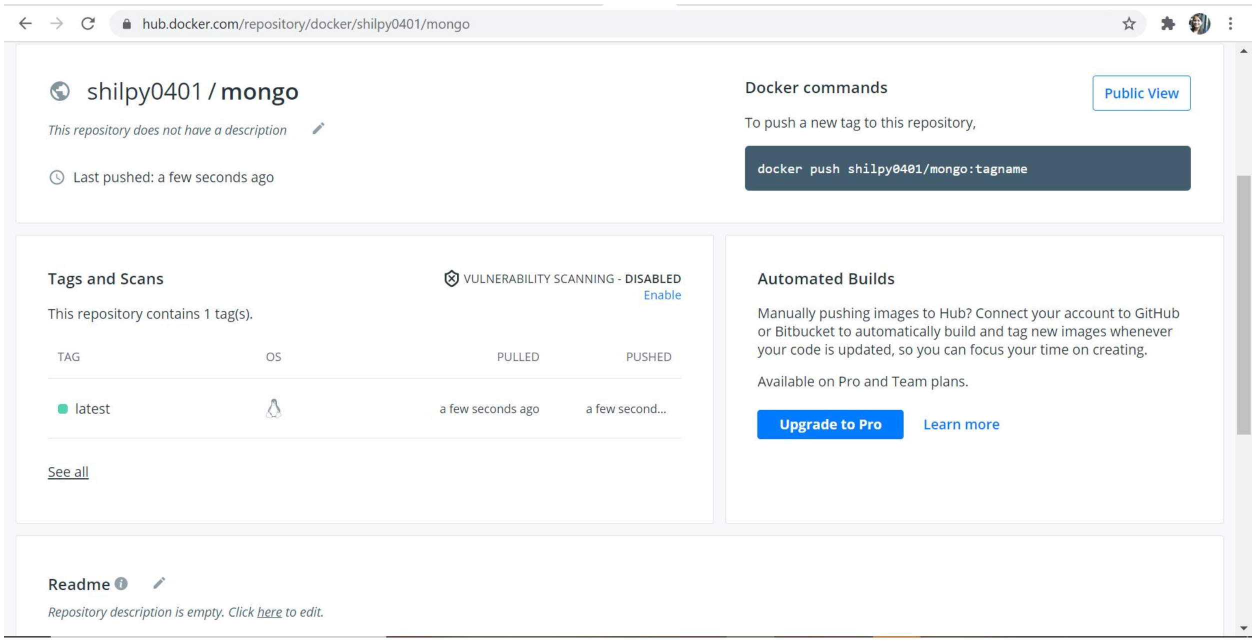


```
root@node05:~/CaseStudy#
root@node05:~/CaseStudy# docker run -d --name DockerCaseStudy -v vol1:/root shilpy0401/mongo:latest
3326cf65298abf0dbaa6fb1720128066 2bde071783627dc7ccdc817243ffdadf
```

Push the docker image to your docker hub account so that it can be pulled later



```
root@node05:~/CaseStudy# docker push shilpy0401/mongo
Using default tag: latest
The push refers to repository [docker.io/shilpy0401/mongo]
912c59f6498b: Pushed
6058600ee9e1: Pushed
b6af68d102f3: Pushed
ceaf9e1ebef5: Mounted from lerndevops/openjdk8
9b9b7f3d56a0: Mounted from lerndevops/openjdk8
f1b5933fe4b5: Mounted from lerndevops/openjdk8
latest: digest: sha256:a114543e29fd353213d1dc07b969a4727722bc921fa010e20adad605e025d3e4 size: 1577
root@node05:~/CaseStudy# |
```



hub.docker.com/repository/docker/shilpy0401/mongo

shilpy0401 / mongo

*This repository does not have a description*

Last pushed: a few seconds ago

Docker commands

Public View

To push a new tag to this repository,

```
docker push shilpy0401/mongo:tagname
```

Tags and Scans

VULNERABILITY SCANNING - DISABLED
Enable

This repository contains 1 tag(s).

| TAG | OS | PULLED | PUSHED |
|---|---|---|---|
| latest | | a few seconds ago | a few second... |

See all

Automated Builds

Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available on Pro and Team plans.

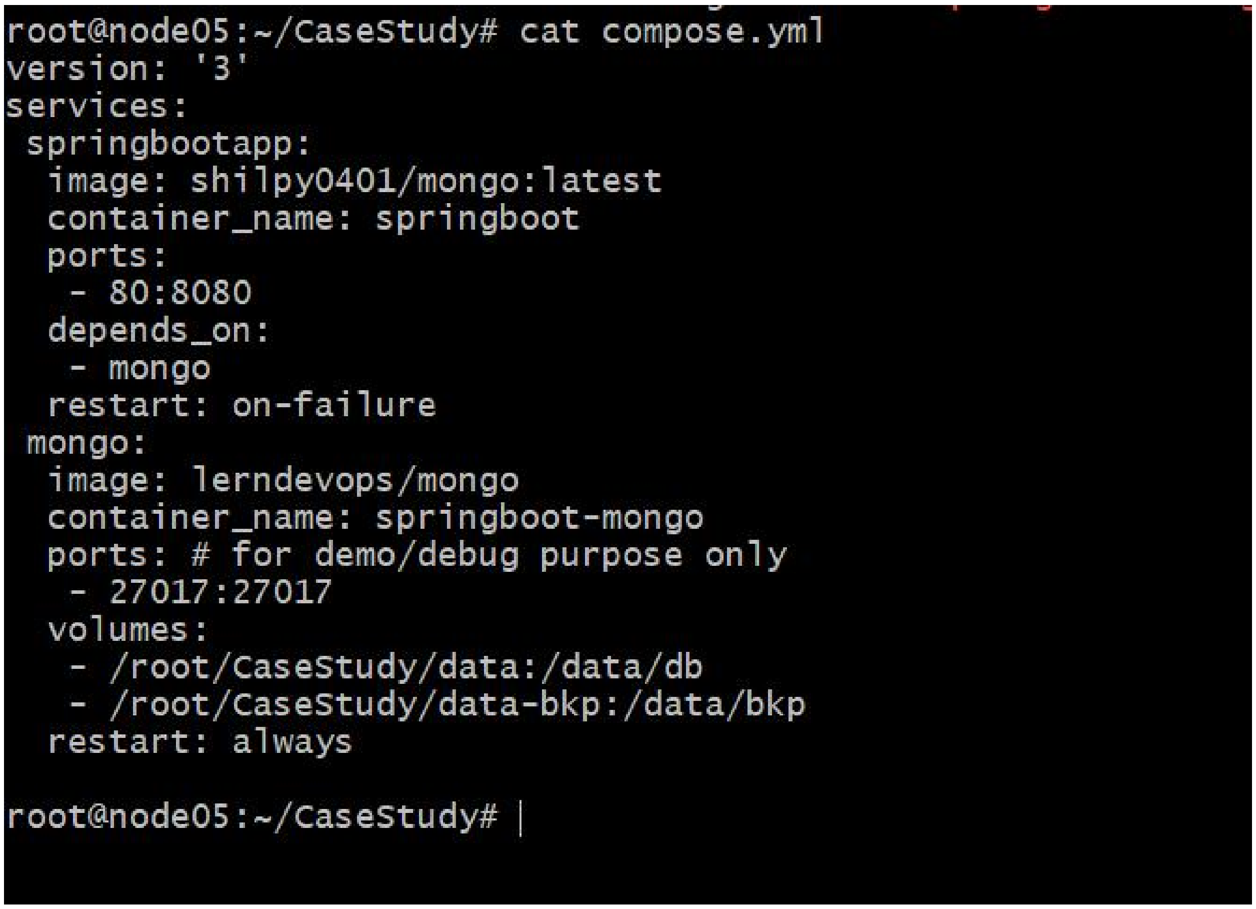
Upgrade to Pro   Learn more

Readme

*Repository description is empty. Click here to edit.*

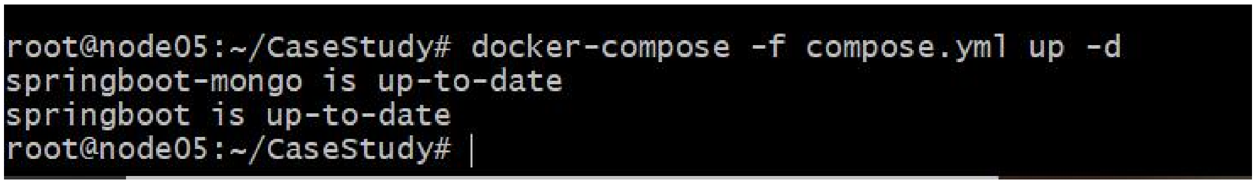
**Create a swarm cluster**

Deploy your firm’s website on the swarm cluster and expose port 80 to access the
website. Also, ensure that the volumes are configured properly so that the source of the files is the same for all the containers of the service

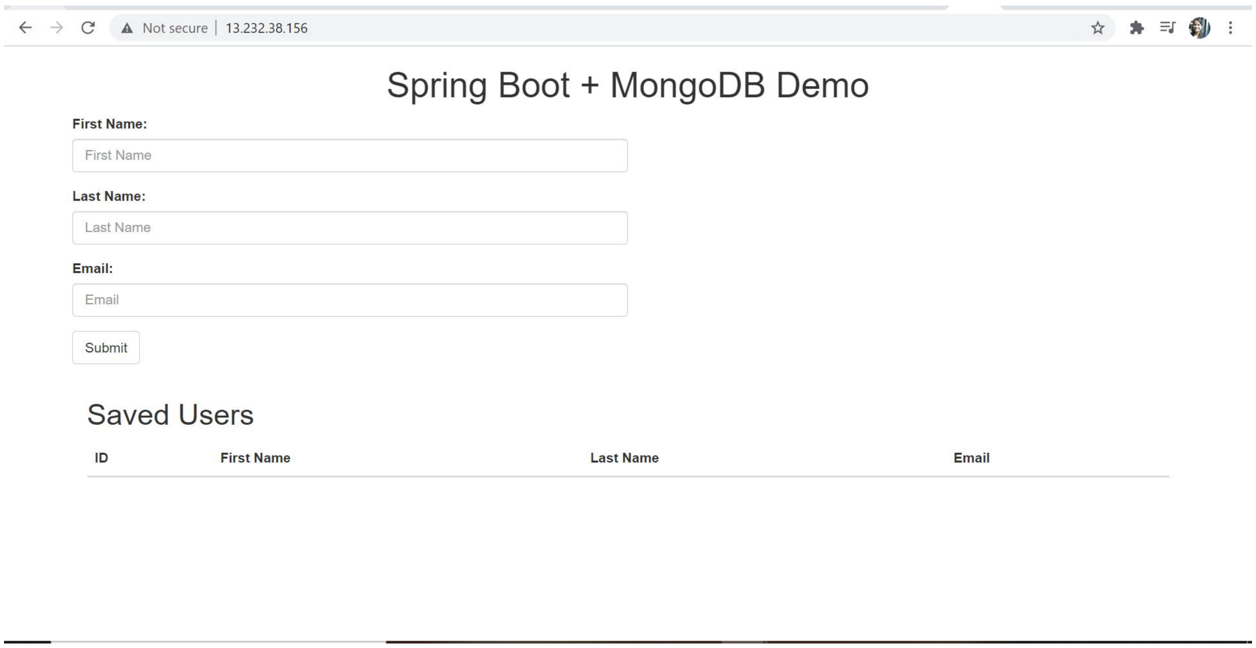


```
root@node05:~/CaseStudy# cat compose.yml
version: '3'
services:
 springbootapp:
  image: shilpy0401/mongo:latest
  container_name: springboot
  ports:
   - 80:8080
  depends_on:
   - mongo
  restart: on-failure
 mongo:
  image: lerndevops/mongo
  container_name: springboot-mongo
  ports: # for demo/debug purpose only
   - 27017:27017
  volumes:
   - /root/CaseStudy/data:/data/db
   - /root/CaseStudy/data-bkp:/data/bkp
  restart: always

root@node05:~/CaseStudy# |
```



```
root@node05:~/CaseStudy# docker-compose -f compose.yml up -d
springboot-mongo is up-to-date
springboot is up-to-date
root@node05:~/CaseStudy# |
```



Spring Boot + MongoDB Demo

First Name:

First Name

Last Name:

Last Name

Email:

Email

Submit

Saved Users

| ID | First Name | Last Name | Email |
|---|---|---|---|