# Name: Sayan Bhattacharjee

MID: M1056174

Course Name: Managing the docker

images 201

Docker Images 201

## Problem Statement:

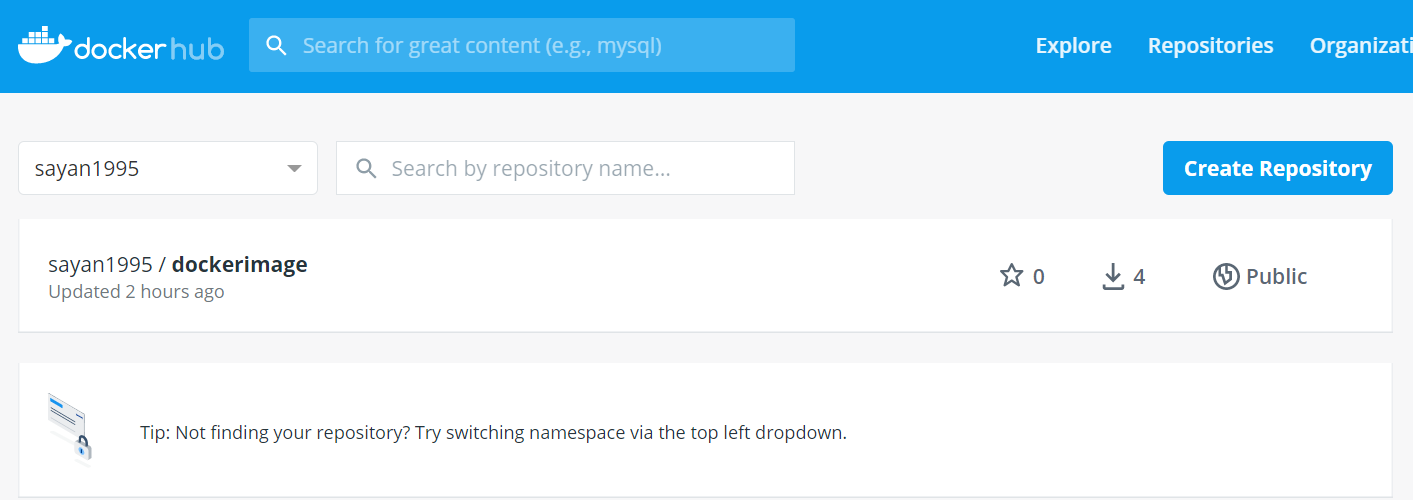
Build a 2-tier java application, Write a Docker file for creating an image & containerize the application. Push the image to Docker hub, deploy it and get all the parts to work together.

## Execution Steps:

For the ease of execution I have downloaded a sample java application from GITHUB.

Account Creation:

Create a Docker account in Docker hub official site (<https://hub.docker.com/>). Created account is used to create a repository and push the Docker images. Now we have to create a repository.



Clone the application from GitHub that has to be dockerized.

In order to do that first open Cloud Lab with given credentials.

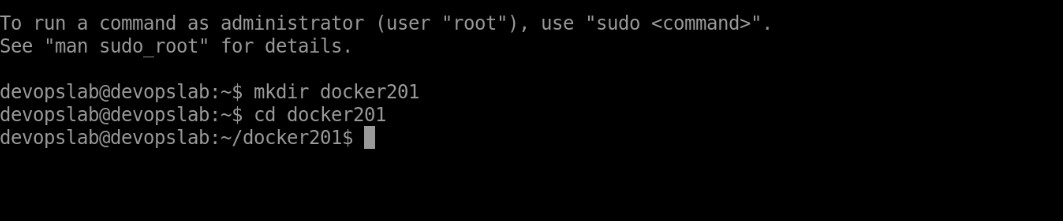
## Docker File:

Docker file is a text document that contains all the commands which assemble to create Docker image. We need to create a Docker file to containerize the application. Docker can build the images by reading instructions from the Docker file.

## Steps to create a Docker file:

1. We need to create a directory first in order to create files and build the container

Commands: **mkdir docker201** to create directory and the type **cd docker201** to get into this newly created directory.



1. Now create a Docker file to write the configuration to build the container. Command: **touch Dockerfile**

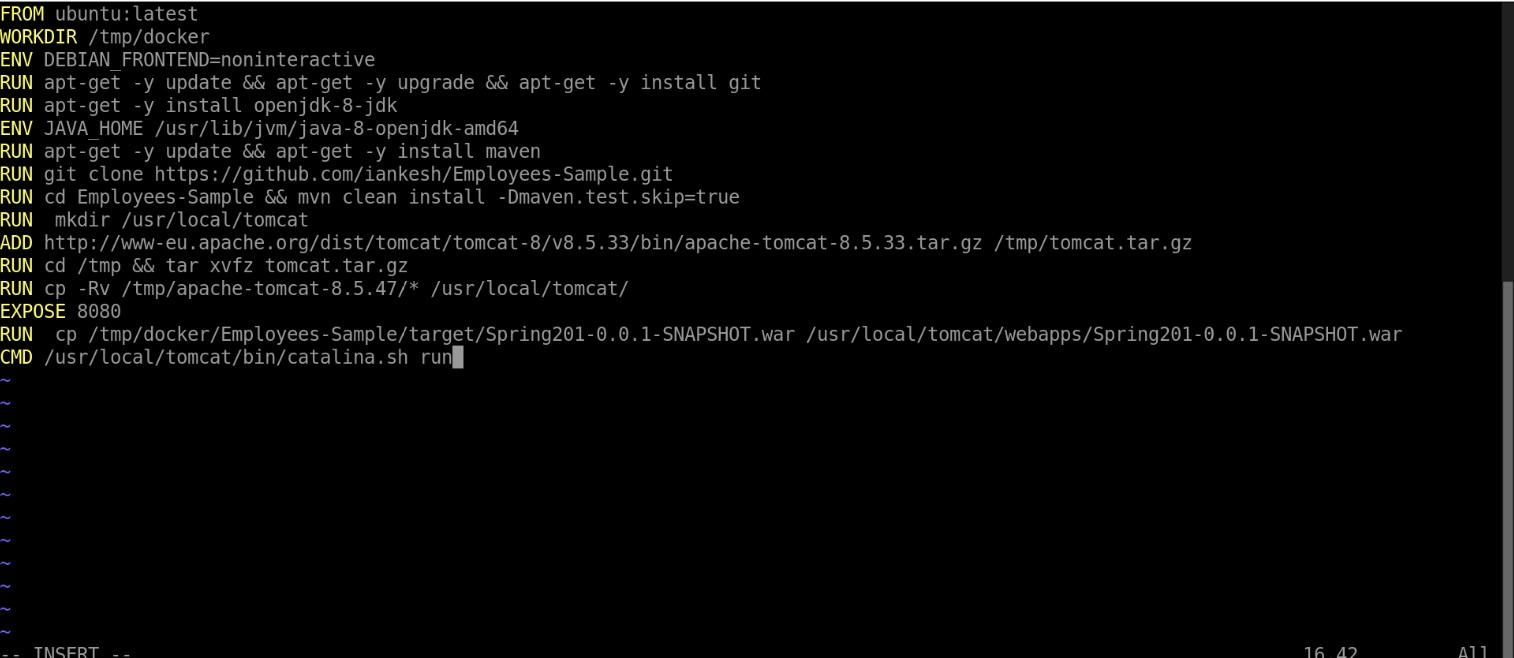
We can see the content of the home directory with **ls** command.



To see/edit the contents of the Dockerfile the below command should be used.

### vi Dockerfile

We have to add the below content in the Dockerfile.



**:wq** is the command to save into the changes into the file.

Let’s understand each command of the Dockerfile.

**FROM**: It is used to pull a public image from Docker-Hub and get that added to the build. In our case, it initializes an Ubuntu instance with the latest configuration.

**WORKDIR**: It is used to set the working directory to run all the commands from the Dockerfile.

**RUN**: It is used to execute any command on the top of the image and commits the result.

**ENV**: It is used to set the home path of the different application. In our case, we are setting the java path using this command.

**ADD**: It is used to copy new files, directories or downloads from remote URLs.

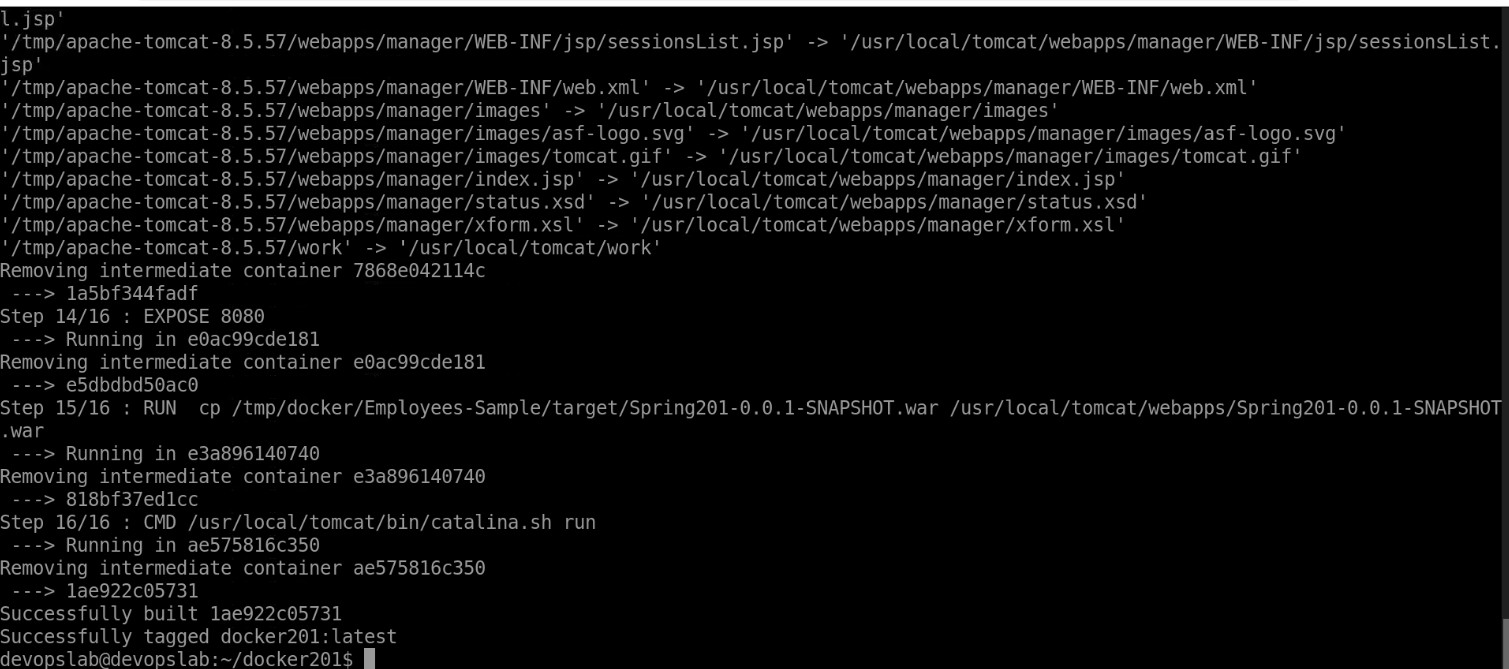
**EXPOSE**: It ensures the container that the application listens to the specific port at runtime.

**CMD**: It is used to provide defaults in running container. It must be used only once in the Dockerfile, if used more than one time, only last CMD will come into effect.

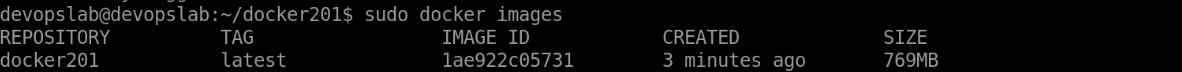
## Building the Docker Image using the Dockerfile

Command: **docker build –t docker201**

Note: Here docker build is the command and –t docker201 is the tag that we are assigning to the image.



To check if the docker image is created properly we use the following command Command: **docker images**



## Pushing a Docker Image into Docker hub

Now that we have created our image the next step is to push the data into our docker-hub. The command for pushing the image is given below.

### Command: docker tag docker201 sayan1995/dockerimage and then type the command: docker push sayan1995/dockerimage.



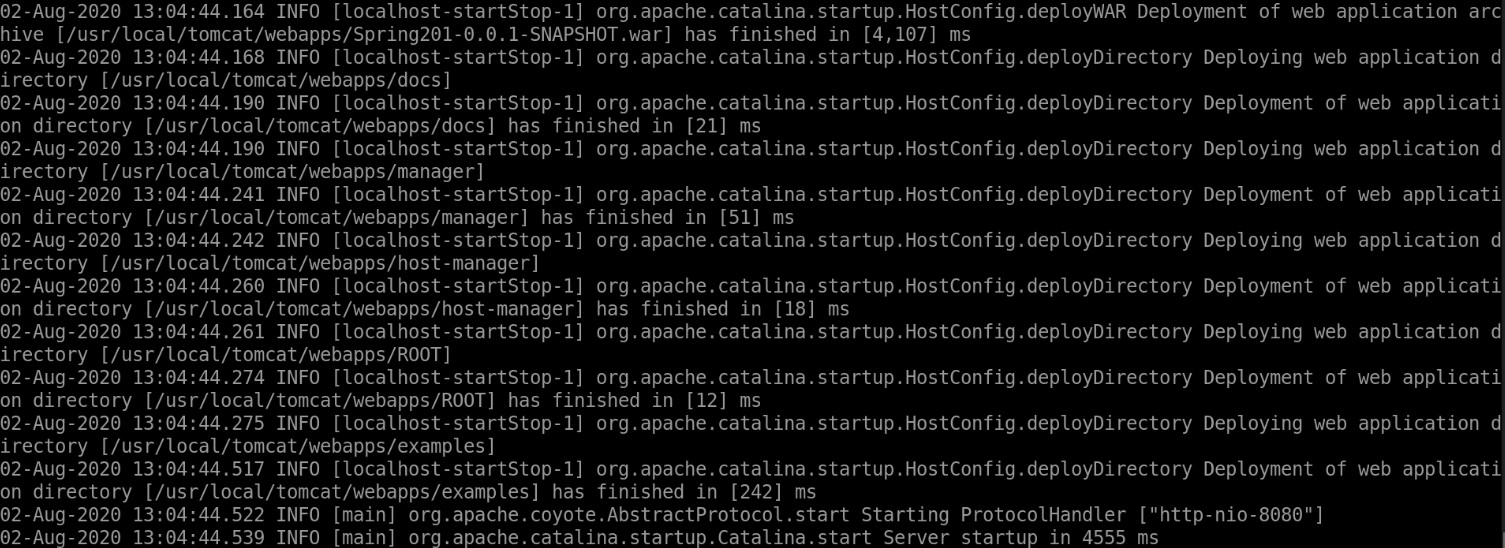
## Running a Docker container:

We have our image ready. The command used to run this image to create and start the container is given below.

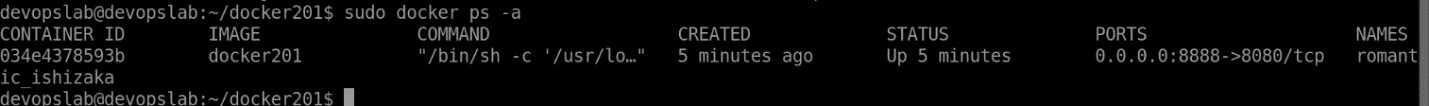
### Command: docker run –it –p 8888:8080 docker201

( **docker run** is the command use to create and run the docker container. **–it** is the interactive mode.

**–p 8888:8080** exposes the container port to the host port. Here we have exported 8080 port of the container to the 8888 port of the host machine. And **docker201** is the image name that is created.)

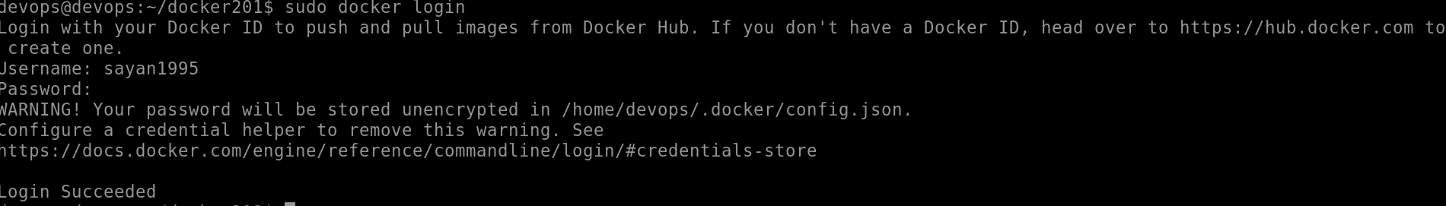


Once the docker container is up and running it can be displayed with the below command. Command: **docker ps –a**



## Adding the database in the docker.

Login into your docker account with the created id and password. The command to login is given below. Command: **docker login**

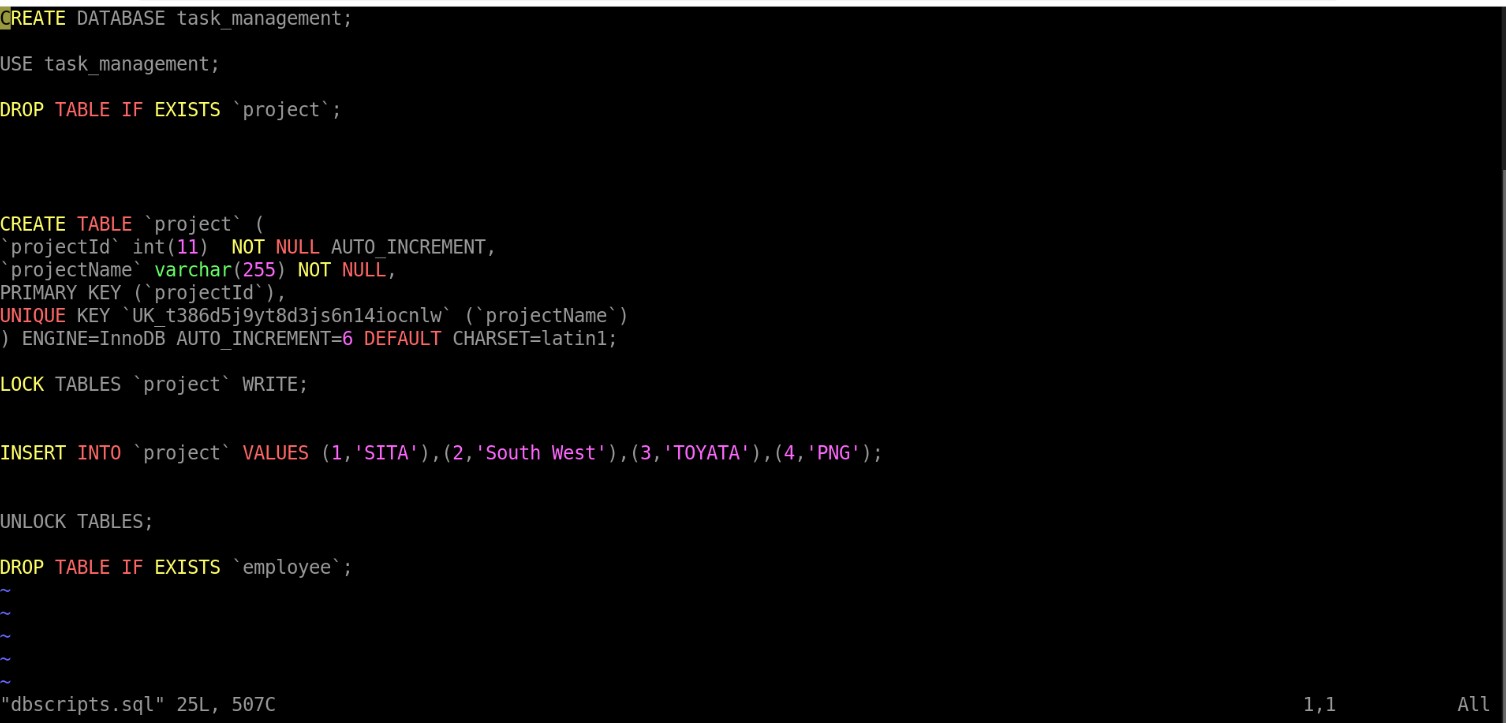


After successfully logging into the docker account. Now we have to create the db scripts.

Create a sql file with the command: **touch dbscripts.sql**

Now to edit the sql file use command: **vi dbscripts.sql**

Now the below content should be added into the dbscripts.sql file



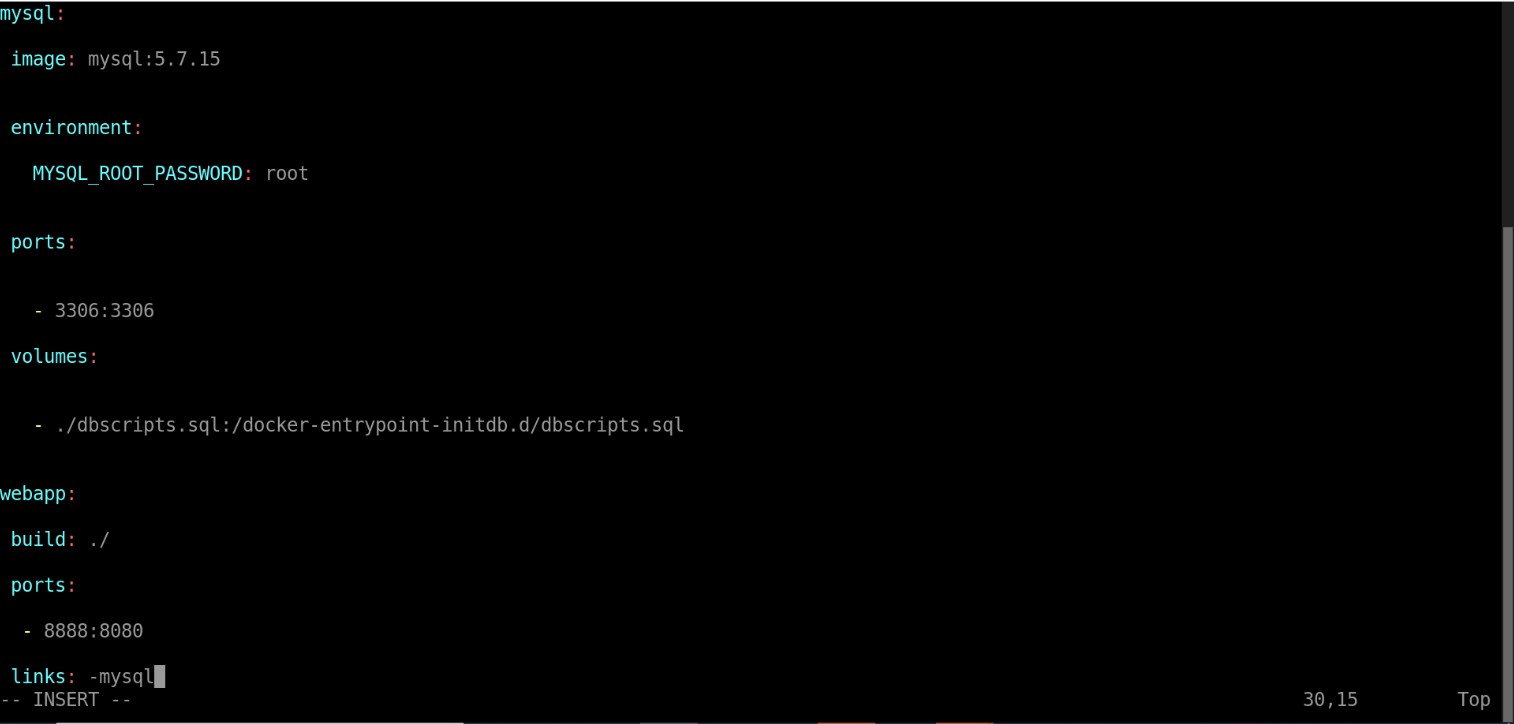
(When the suitable instructions are inserted hit <esc> and type :wq that will save the file)

## Running the docker container

Now we have to create a docker-compose file by using the below command. Command: **touch docker-compose.yml**

To edit the docker-compose file we use the below command. Command: **vi docker-compose.yml**

Add the below shown content into it

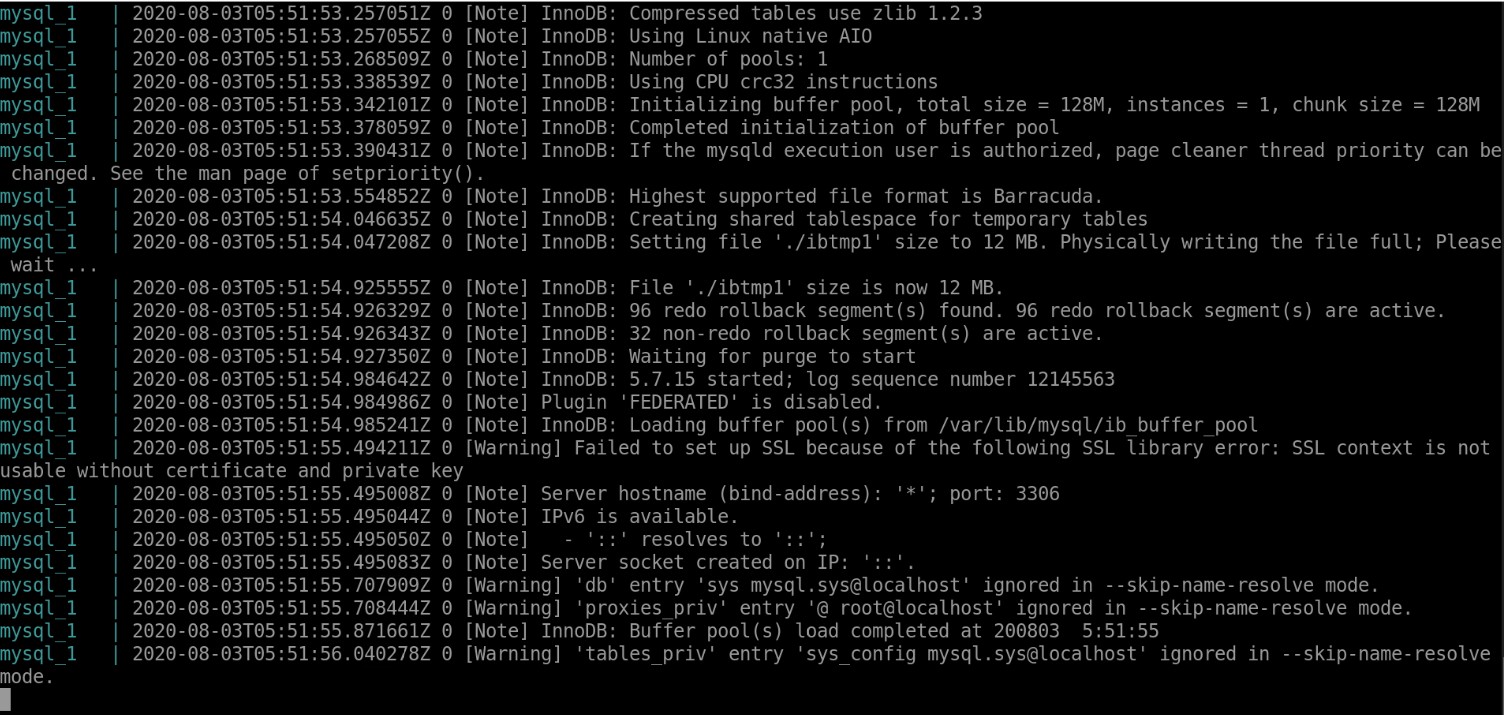


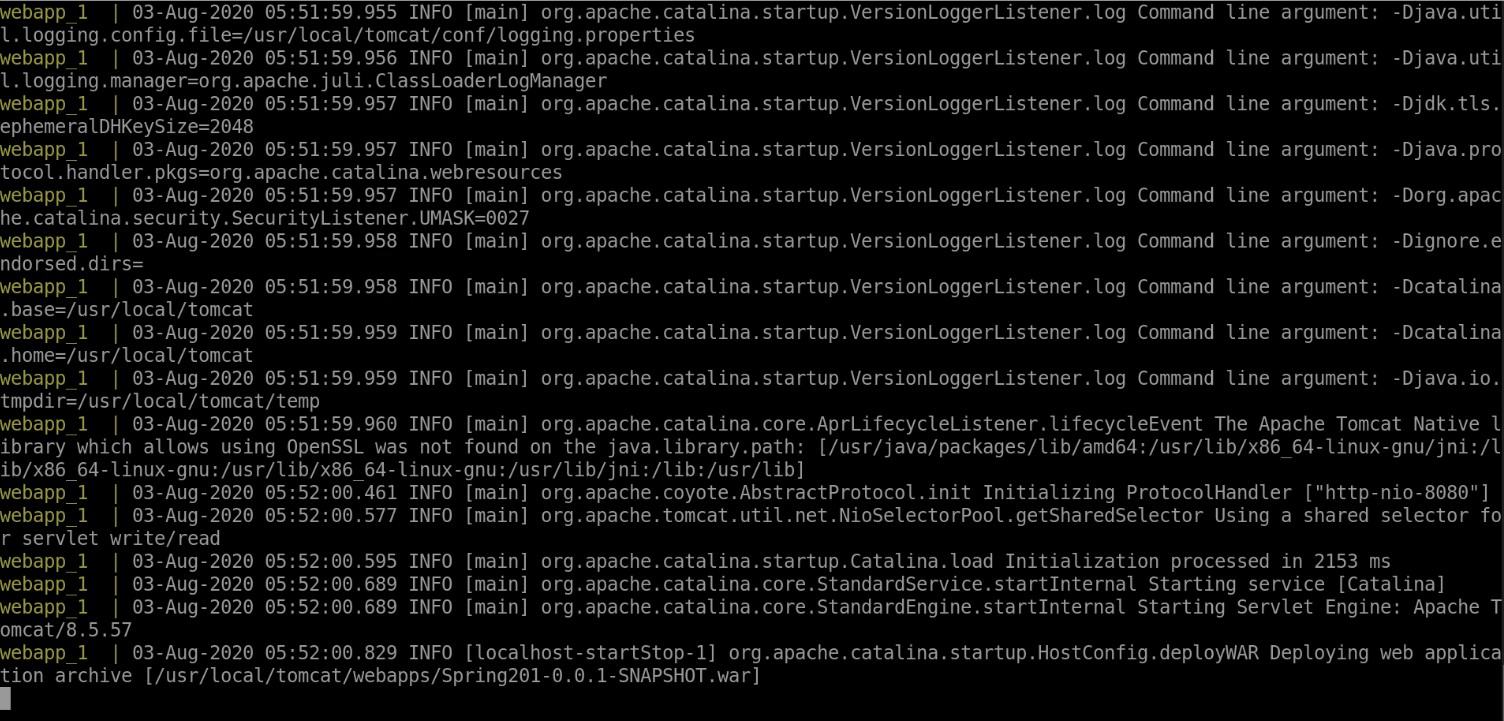
(When the suitable instructions are inserted hit <esc> and type :wq that will save the file)

Command to make the container up is **docker-compose up**

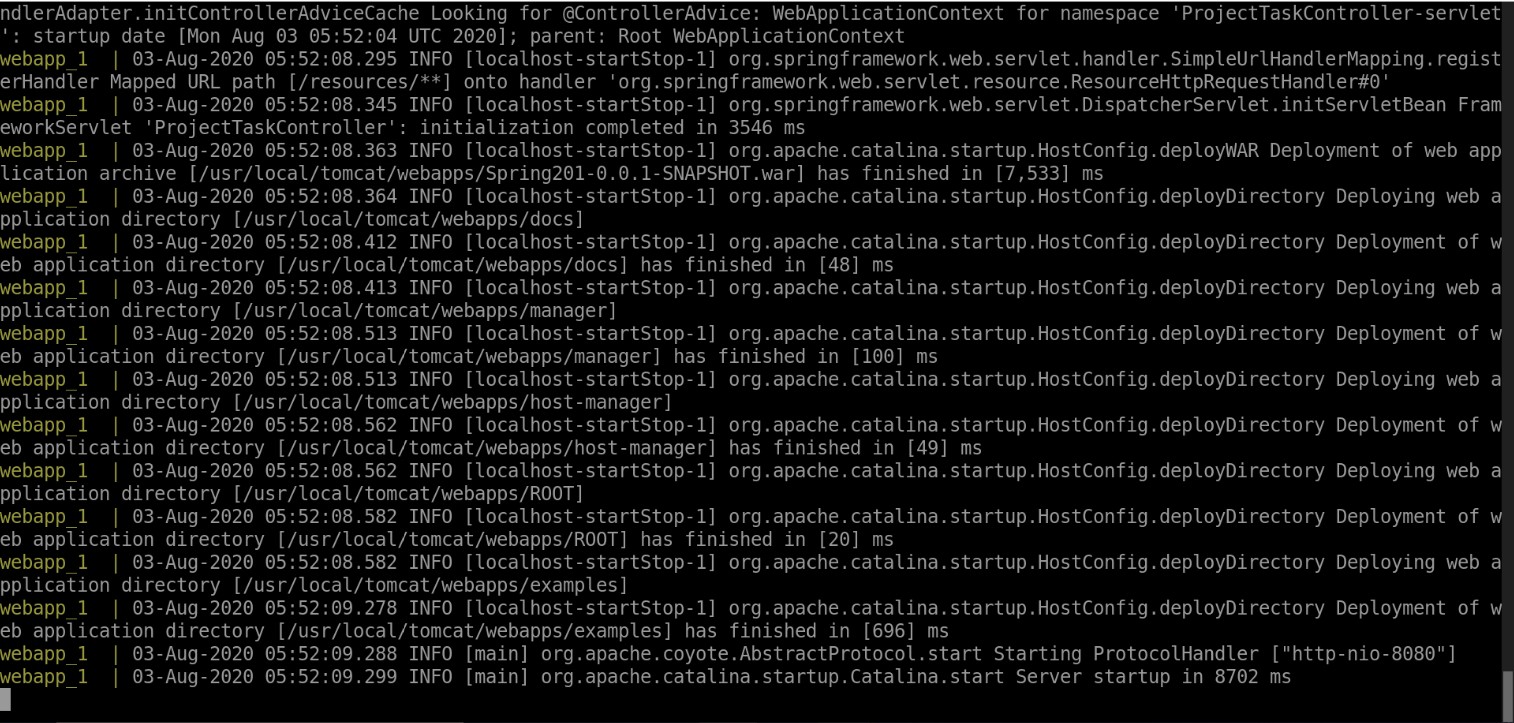
Similarly to make it down the command is **docker-compose down**

To start we use the **docker-compose up** command









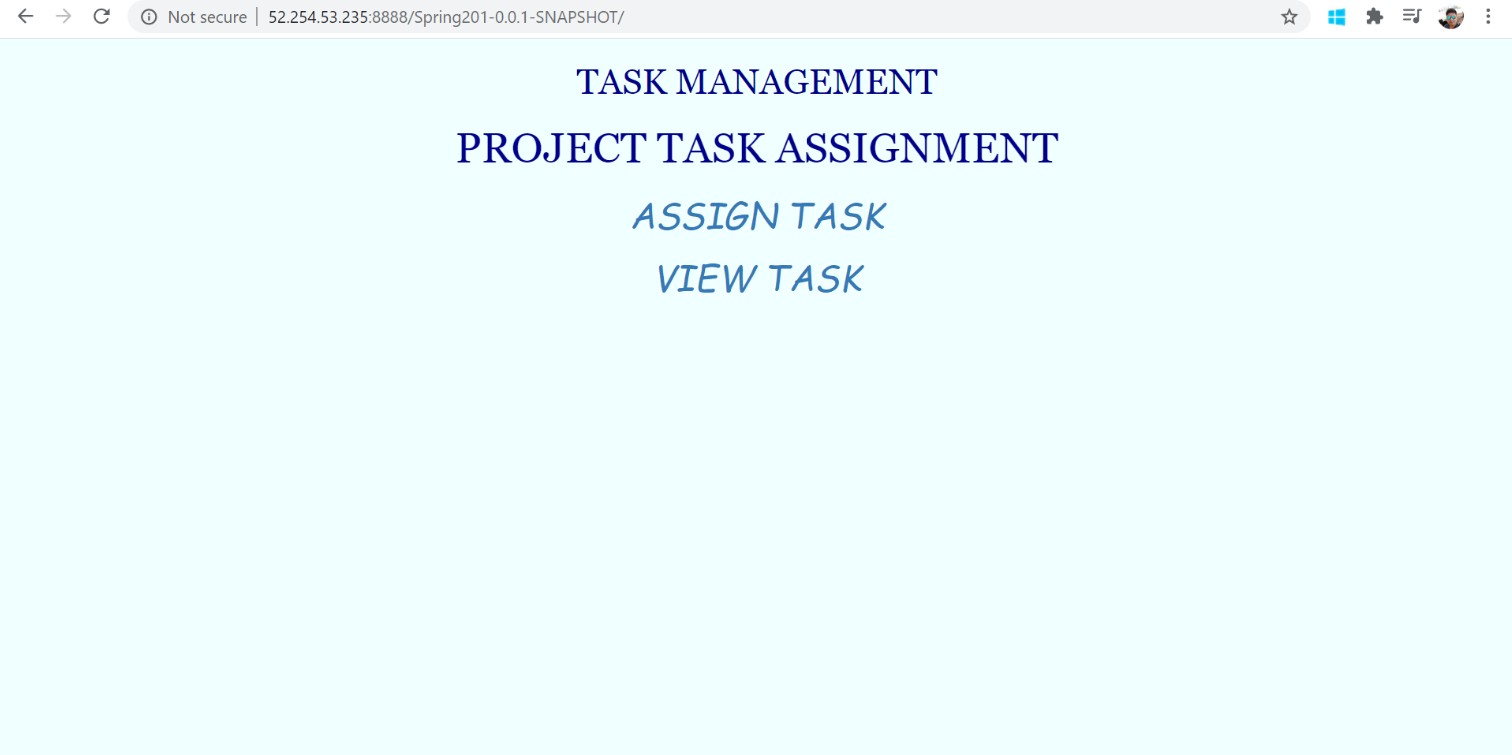
Now we can safely start the application.

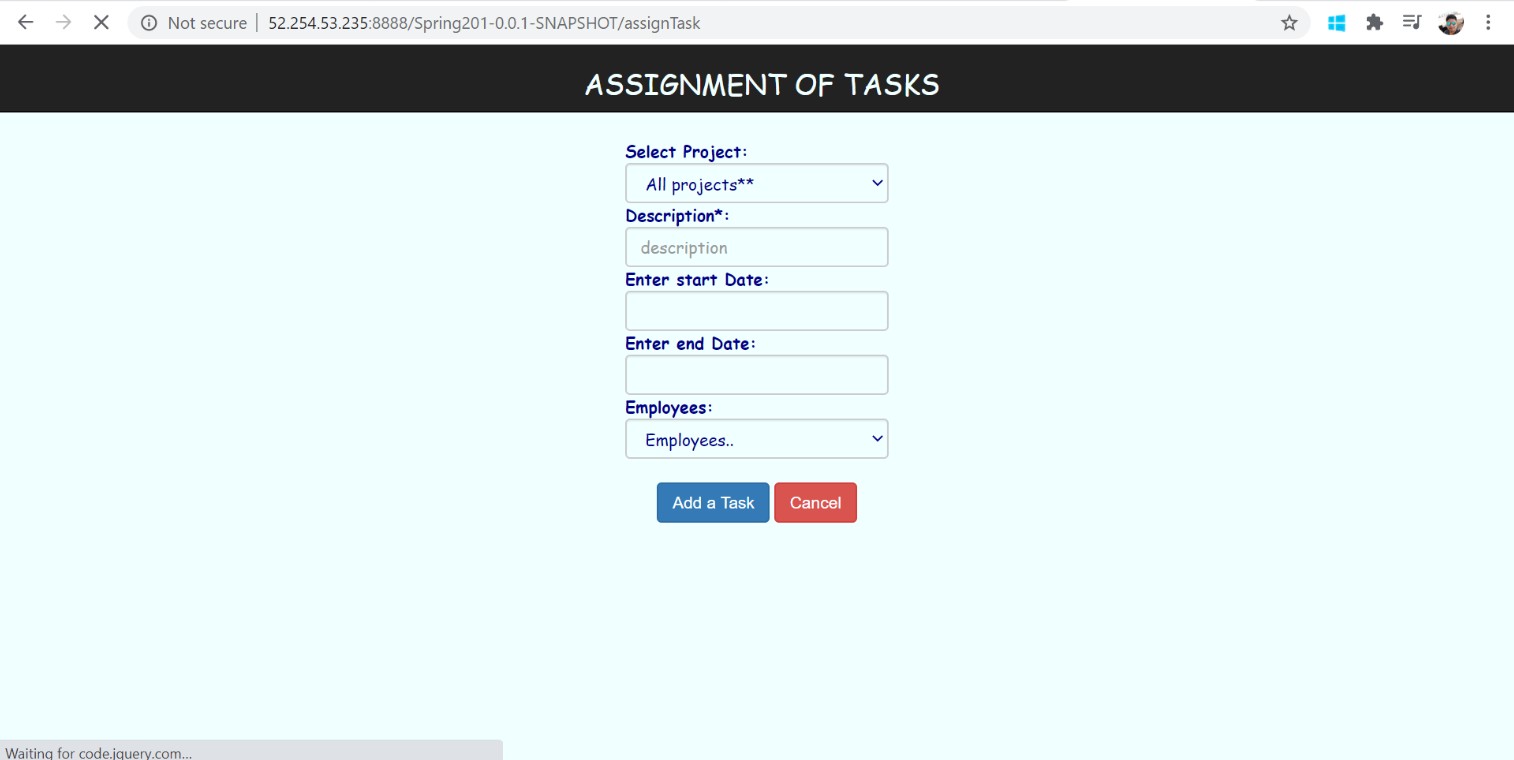
But in order to start we have to know the hostname in which the application can be started. Command to know the hostname: **curl ifconfig.me**

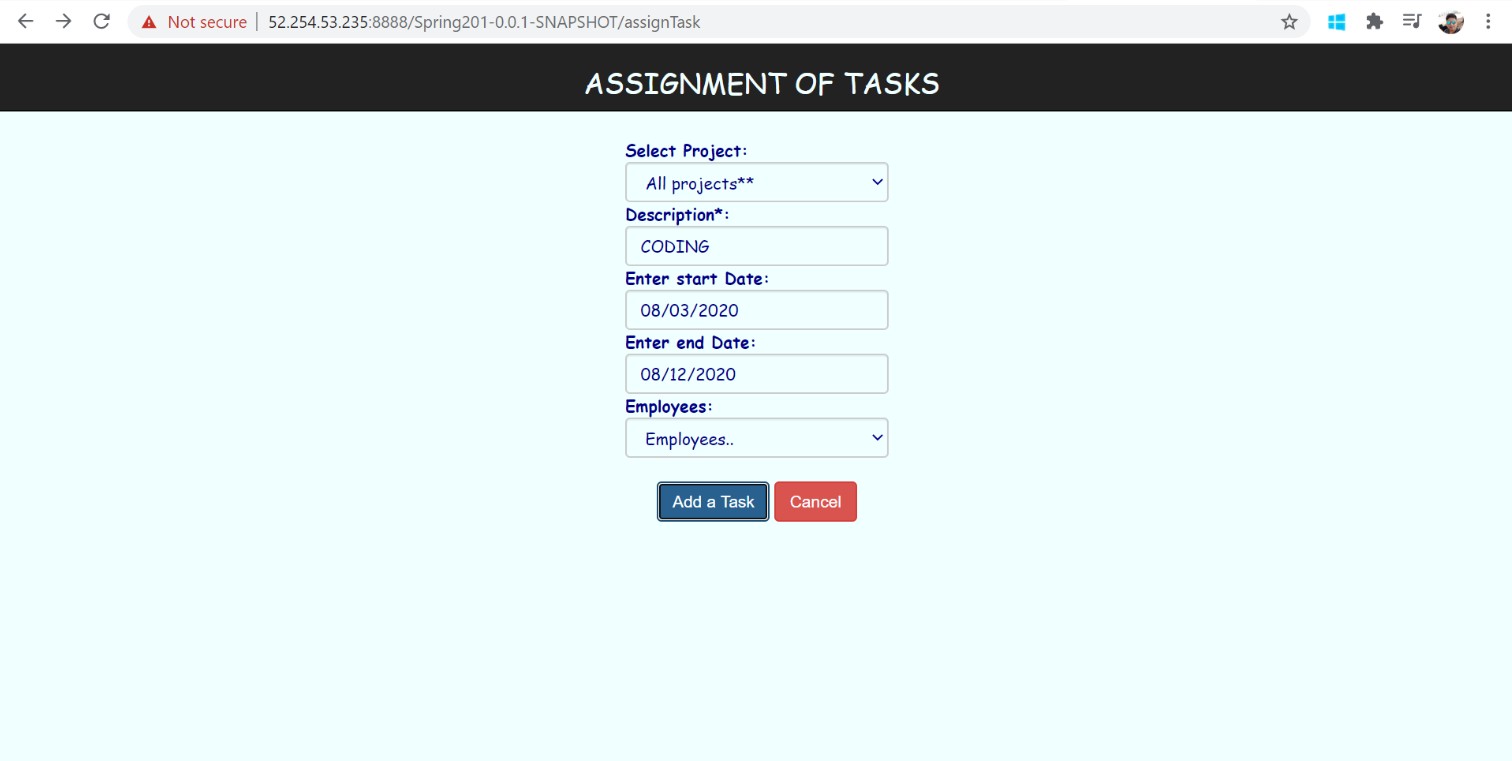


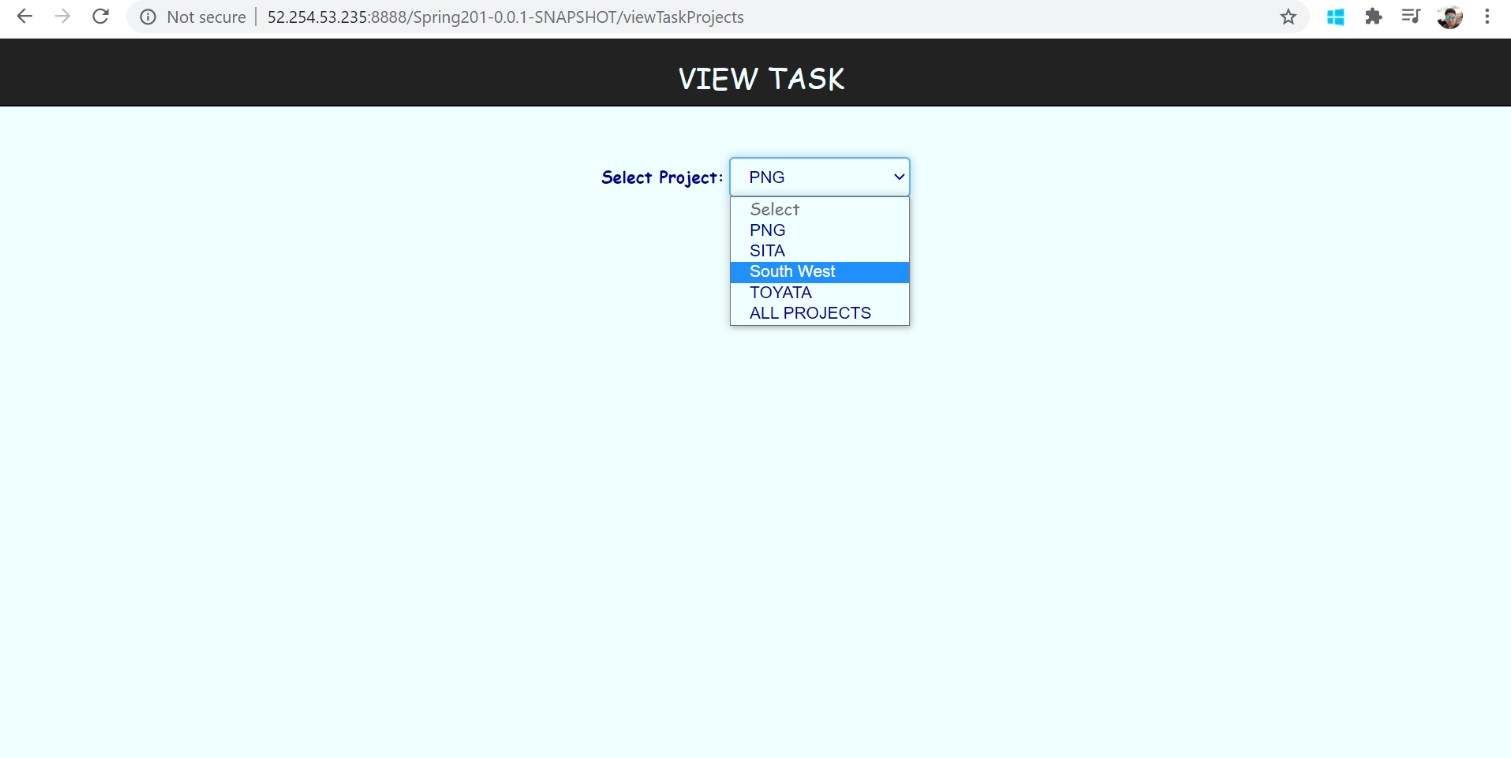
My hostname is: **52.254.53.235**

The URL address which is used to access the application is: **http:// 52.254.53.235:8888/Spring201- 0.0.1- SNAPSHOT/**









# Thank You!