Assignment 1 (Docker)

Big Data Mining in Healthcare

By Group 11:

Yashraj, 2018422

Nitesh, 2018400

Karan, 2018394

Assignment using Single instance of container:-

Docker link:- https://hub.docker.com/r/nitesh18400/crudu

Download command:- sudo docker pull nitesh18400/crudu:1.4

Assignment using Multiple instances of containers (BONUS):-

Github link:- https://github.com/nitesh18400/CRUD

Question 1:

We have installed MySQL in our docker container but upon restarting the system, the MySQL service doesn't run automatically. In case if it doesn't work we have to manually restart the MySQL service by executing the following command after starting a new instance of the shell:

docker ps **(To get Container-ID)**

docker exec -it Container-ID sh **(To start a new instance of the shell by entering Container-ID that we got above)**

service mysql restart **(To restart MySQL service which doesn't start automatically)**

```
Terminal                    Terminal                    Terminal
┌─[nitesh@nitesh-Vostro-3458]─[~]
└──╼ $docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED            STATUS             PORTS                                  NAMES
af8f9af041cc   nitesh18400/crudu:1.2  "/bin/sh -c 'python3…"   About an hour ago  Up About an hour   3306/tcp, 0.0.0.0:8000->8000/tcp       vigorous_ard
inghelli
┌─[nitesh@nitesh-Vostro-3458]─[~]
└──╼ $
```

```
┌─[nitesh@nitesh-Vostro-3458]─[~]
└──╼ $docker exec -it af8f9af041cc sh
# service mysql restart
 * Stopping MySQL database server mysqld                                                    [ OK ]
 * Starting MySQL database server mysqld                                                    [ OK ]
```

<u>Commands Used for MySQL Database (mydb):</u>

#mysql **(To start MySQL)**

> create database mydb; **(To create a database with name mydb)**

```
# mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.23-0ubuntu0.20.04.1 (Ubuntu)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| Products           |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.08 sec)

mysql> create database mydb;
```

> use mydb; (**For using “mydb” database**)

> CREATE TABLE Persons (PersonID int, LastName varchar(255), FirstName varchar(255), Address varchar(255), City varchar(255));
(To create a table with name Persons)

> INSERT INTO Persons (PersonID, LastName, FirstName, Address, City) VALUES ('101', 'Jaiswal', 'Karan', 'WZ-101', 'Delhi');

> INSERT INTO Persons (PersonID, LastName, FirstName, Address, City) VALUES ('102', 'Chawla', 'Harshit', 'WZ-102', 'Delhi');
(To Insert rows into table)

> select * from Persons;
(To show contents of the table)

```
mysql> use mydb
Database changed
mysql> CREATE TABLE Persons (
    ->     PersonID int,
    ->     LastName varchar(255),
    ->     FirstName varchar(255),
    ->     Address varchar(255),
    ->     City varchar(255)
    -> );
Query OK, 0 rows affected (0.66 sec)

mysql> show tables;
+----------------+
| Tables_in_mydb |
+----------------+
| Persons        |
+----------------+
1 row in set (0.01 sec)

mysql> INSERT INTO Persons (PersonID, LastName, FirstName, Address, City)
    -> VALUES ('101', 'Jaiswal', 'Karan', 'WZ-101', 'Delhi');
Query OK, 1 row affected (0.16 sec)

mysql> INSERT INTO Persons (PersonID, LastName, FirstName, Address, City)
    -> VALUES ('102', 'Chawla', 'Harshit', 'WZ-102', 'Delhi');
Query OK, 1 row affected (0.14 sec)

mysql> select * from Persons;
+----------+----------+-----------+---------+-------+
| PersonID | LastName | FirstName | Address | City  |
+----------+----------+-----------+---------+-------+
|      101 | Jaiswal  | Karan     | WZ-101  | Delhi |
|      102 | Chawla   | Harshit   | WZ-102  | Delhi |
+----------+----------+-----------+---------+-------+
2 rows in set (0.03 sec)
```

> SELECT COUNT(PersonID) FROM Persons; **(To count number of persons)**

```
mysql> SELECT COUNT(PersonID)
    -> FROM Persons;
+----------------+
| COUNT(PersonID) |
+----------------+
|              2 |
+----------------+
1 row in set (0.03 sec)

mysql>
```

> UPDATE Persons SET Address = 'W-1010101', City= 'Faridabad' WHERE PersonID = 101; **(To update address and city of a particular person with given PersonID)**

```
mysql> UPDATE Persons
    -> SET Address = 'W-1010101', City= 'Faridabad'
    -> WHERE PersonID = 101;
Query OK, 1 row affected (0.17 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from Persons;
+----------+----------+-----------+-----------+-----------+
| PersonID | LastName | FirstName | Address   | City      |
+----------+----------+-----------+-----------+-----------+
|      101 | Jaiswal  | Karan     | W-1010101 | Faridabad |
|      102 | Chawla   | Harshit   | WZ-102    | Delhi     |
+----------+----------+-----------+-----------+-----------+
2 rows in set (0.00 sec)

mysql>
```

> DELETE FROM Persons WHERE PersonID = 101; **(To delete a particular person with given PersonID)**

```
mysql> DELETE FROM Persons WHERE PersonID = 101;
Query OK, 1 row affected (0.16 sec)

mysql> select * from Persons;
+----------+----------+-----------+---------+-------+
| PersonID | LastName | FirstName | Address | City  |
+----------+----------+-----------+---------+-------+
|      102 | Chawla   | Harshit   | WZ-102  | Delhi |
+----------+----------+-----------+---------+-------+
1 row in set (0.00 sec)

mysql>
```

Question 2:

We have installed MongoDB in our docker container but upon restarting the system, the MongoDB service doesn't run automatically. Incase if it doesn't work we have to manually restart the MongoDB service by executing the following command after starting a new instance of the shell:

docker ps **(To get Container-ID)**

docker exec -it Container-ID sh **(To start a new instance of the shell by entering Container-ID that we got above)**

cd ..

service mongodb stop

rm /var/lib/mongodb/mongod.lock

mongod --repair --dbpath /var/lib/mongodb

mongod --fork --logpath /var/lib/mongodb/mongodb.log --dbpath /var/lib/mongodb

service mongodb start

#mongo **(To start MongoDB)**

Commands Used:

> use mgdb ##**(To create/use database with name mgdb)**

> db.createCollection('tables') ##(**Create Table "tables"**)

> db.tables.insert({"product id":1,"Product_name":"Yashraj","price":1000})

> db.tables.insert({"product id":2,"Product_name":"Nitesh","price":10000}) **##(Insert rows)**

> db.tables.find().count() **##(Count number of rows)**

> db.tables.find() **##(Print rows)**

> db.tables.update({"product id":1},{"Product id":1,"Product_name":"Nit","price":10},{upsert:true }) **##(For Updating)**

> db.tables.find() **##(For Printing rows)**

> db.tables.remove({ "product id": 2 }) **##(Removing row having "product id" = 2)**

> db.tables.find() **##(For Printing rows)**

In this Question, we implement all CRUD operations command and the screenshot is given below:-

```
>
> use mgdb
switched to db mgdb
> db.createCollection('tables')
{ "ok" : 1 }
> db.tables.insert({"product id":1,"Product_name":"Yashraj","price":1000})
WriteResult({ "nInserted" : 1 })
> db.tables.insert({"product id":2,"Product_name":"Nitesh","price":10000})
WriteResult({ "nInserted" : 1 })
> db.tables.find().count()
2
> db.tables.find()
{ "_id" : ObjectId("60391aaa900a3bdae10201ba"), "product id" : 1, "Product_name" : "Yashraj", "price" : 1000 }
{ "_id" : ObjectId("60391aaf900a3bdae10201bb"), "product id" : 2, "Product_name" : "Nitesh", "price" : 10000 }
> db.tables.update({"product id":1},{"Product id":1,"Product_name":"Nit","price":10},{upsert:true })
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.tables.find()
{ "_id" : ObjectId("60391aaa900a3bdae10201ba"), "Product id" : 1, "Product_name" : "Nit", "price" : 10 }
{ "_id" : ObjectId("60391aaf900a3bdae10201bb"), "product id" : 2, "Product_name" : "Nitesh", "price" : 10000 }
> db.tables.remove({ "product id": 2 })
WriteResult({ "nRemoved" : 1 })
> db.tables.find()
{ "_id" : ObjectId("60391aaa900a3bdae10201ba"), "Product id" : 1, "Product_name" : "Nit", "price" : 10 }
>
```

Question 3 and Question 4:

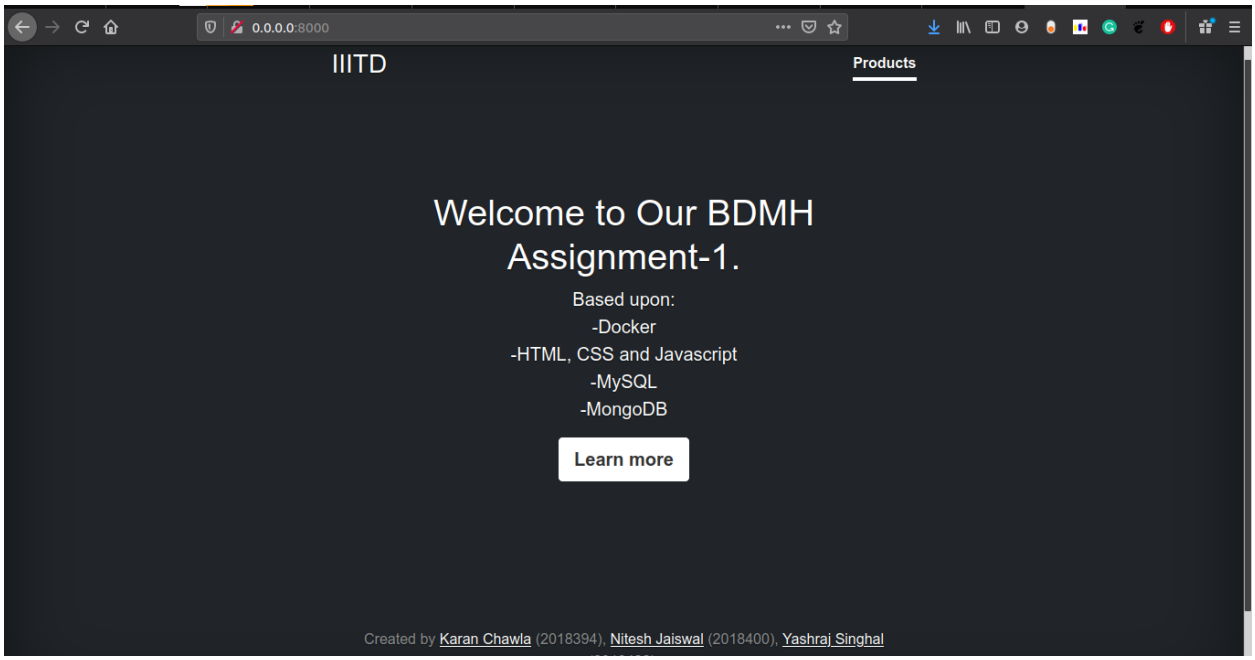
Using - Django and Mysql Stack

Command For Running Container

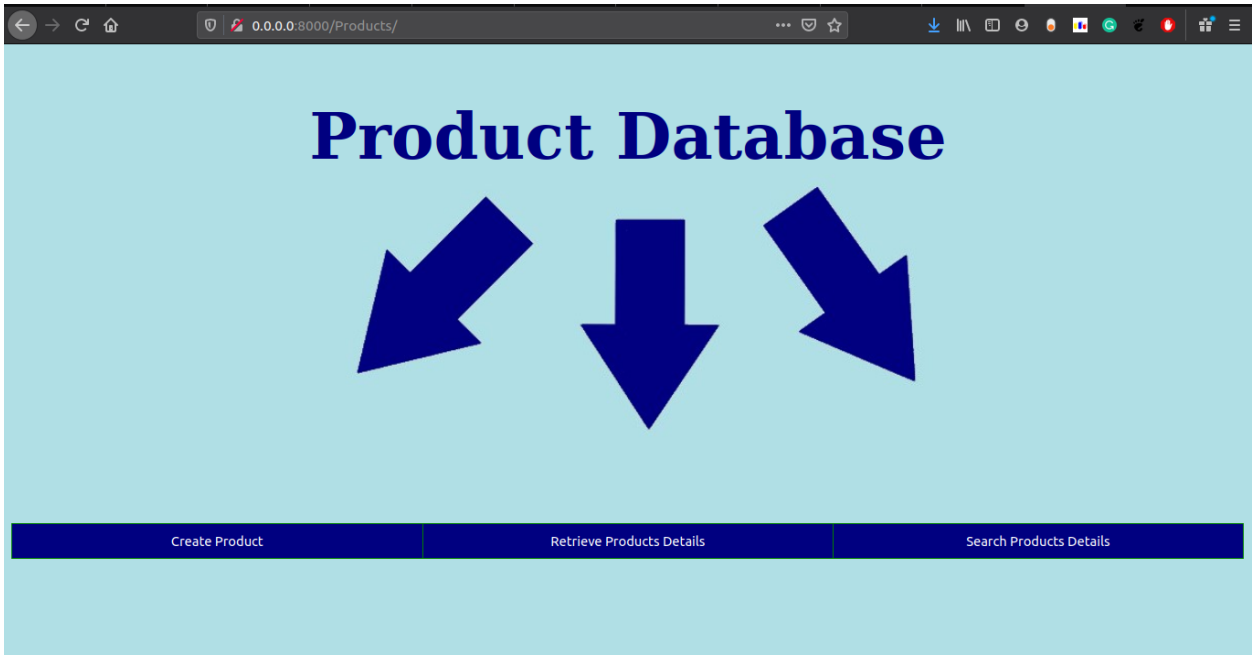
1. Open new terminal session
2. docker pull docker pull nitesh18400/crudu:1.4
3. docker run -p 8000:8000 nitesh18400/crudu:1.4
4. Open another terminal session
5. docker ps (Note Container_id)
6. docker exec -it Noted_Container_id sh
7. service mysql restart

User Interface of Websites pages developed

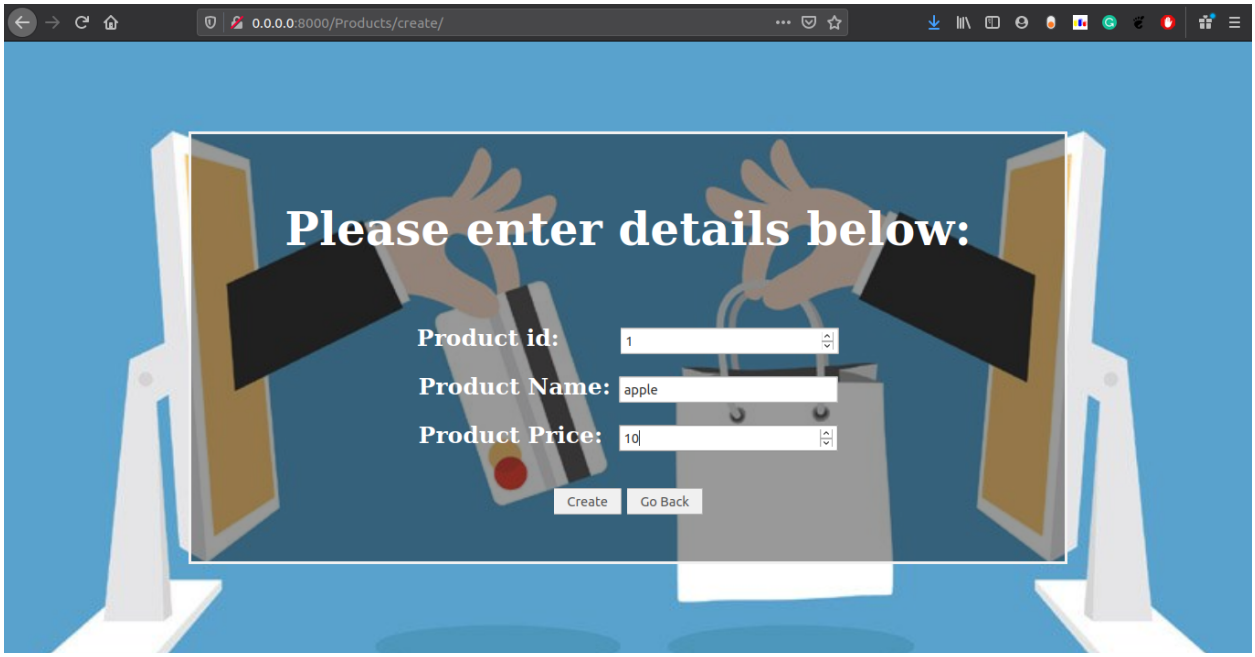
1. Homepage



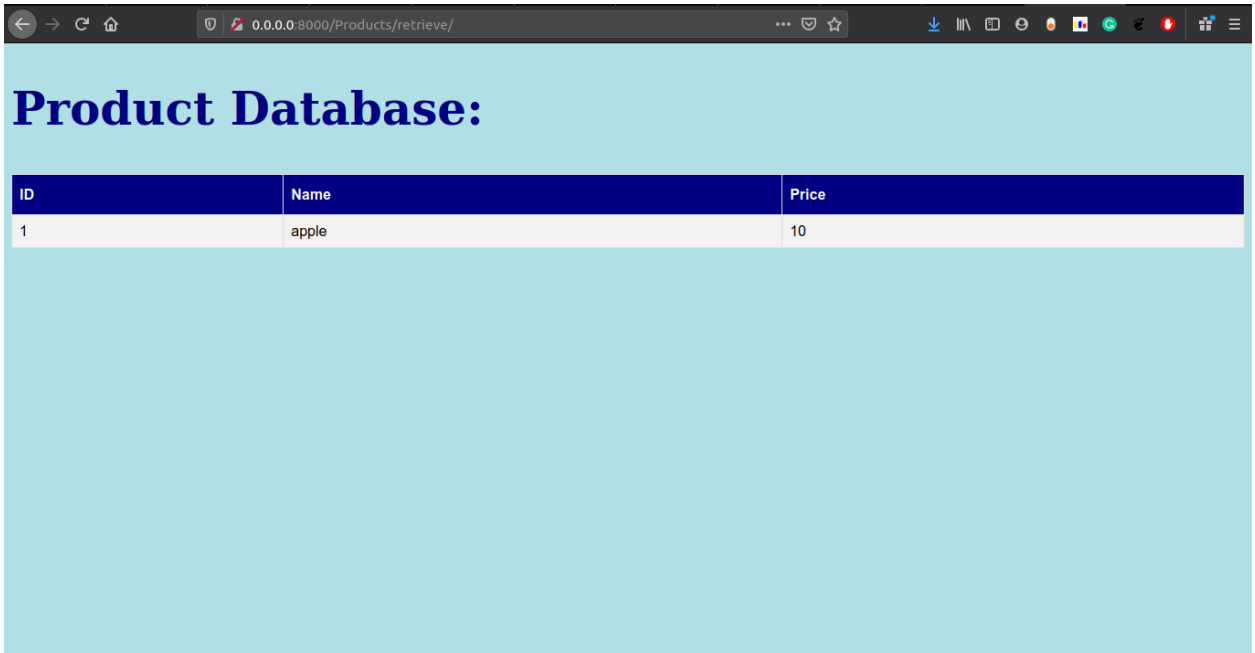
2. Products Page



3. Create Product Page (Add Products to Database)



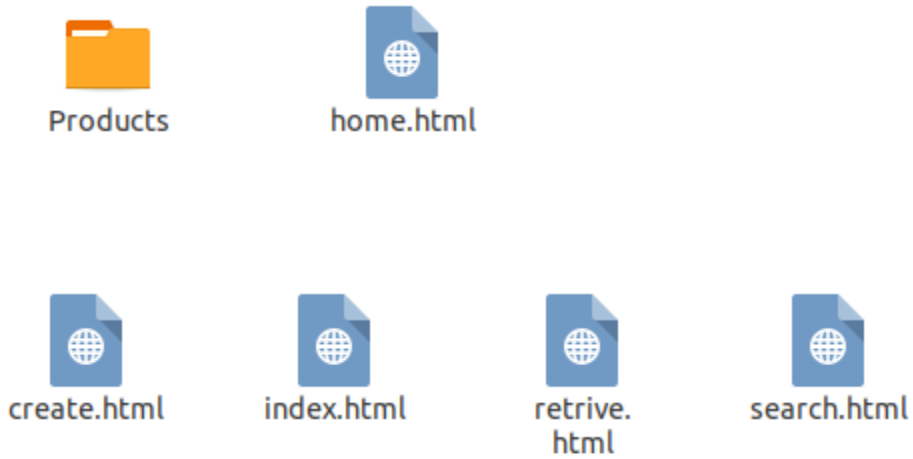
4. Product List Page (Retrieve Products from Database)



5. Search Product Page (Search a Product from Database using id)

HTML files:-

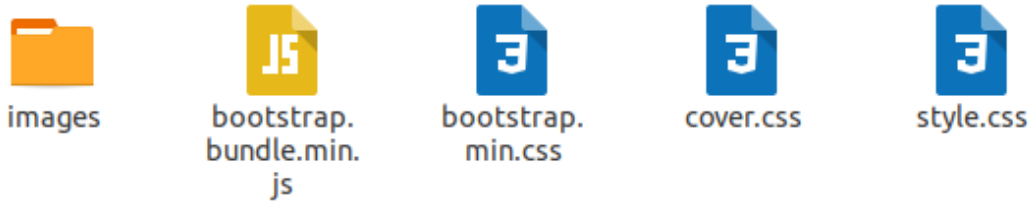
- home.html **(To create a webpage for Main Home Page)**
- create.html **(To create a webpage for Creating/Adding new product into the database)**
- index.html **(Web Page for Index of all pages)**
- search.html **(Web Page to search a product using Product ID in the database)**
- retrieve.html **(To showcase complete data of our database in tabular format)**



CSS/JavaScript Files:-

- cover.css **(CSS file for design of home.html file)**

- style.css **(CSS file for design of create.html file)**
- bootstrap.min.css **(CSS file for design of home.html file)**
- bootstrap.bundle.min **(JS file for home.html file)**



Docker Files (Assignment using Single Container):-

- **Dockerfile** (For Creating Host Environment and Installing required packages)

  Creating Ubuntu Environment

  Installing Required Dependencies for stack

  Hosting Command

Dockerfile ×

Dockerfile > ...

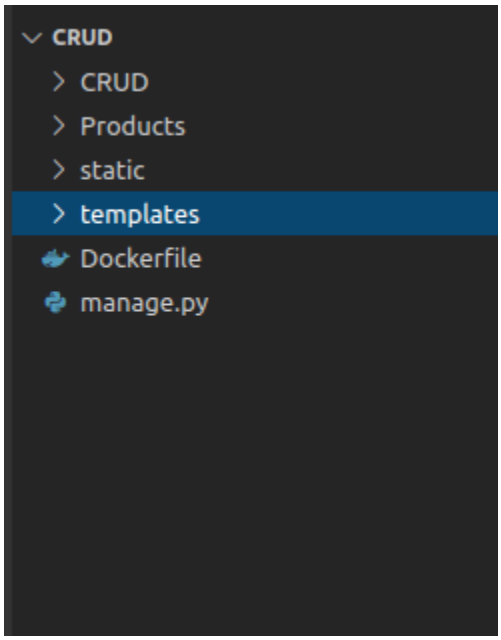
```
FROM ubuntu
ENV PYTHONUNBUFFERED 1

RUN apt-get -y update

ARG DEBIAN_FRONTEND=noninteractive
RUN apt-get install -y software-properties-common
RUN apt-get -y install mysql-server libmysqlclient-dev
RUN apt-get install -y python3-pip
RUN pip3 install --upgrade pip
RUN pip3 install Django==3.1.3 PyMySQL==1.0.2

RUN mkdir /app
WORKDIR /app
COPY . /app/
CMD python3 manage.py makemigrations && python3 manage.py makemigrations && python3 manage.py runserver
# ENTRYPOINT service ssh restart && bash
```

- **Stack File Structure** (For Single Container)

- 

Product -> Django Application

Template -> Containing HTML Files

Static -> Containing CSS and JS Files

Mange.py -> Main Host file for launching of Website

Docker Files (Assignment using Multiple Containers):-

- **Dockerfile** (For Creating Host Environment and Installing required packages)

  Creating Ubuntu Environment

  Installing Required Dependencies for stack mentioned in requirement.txt

  Hosting Command

Dockerfile

```
FROM python:3.9
ENV PYTHONUNBUFFERED 1
RUN mkdir /app
COPY requirement.txt /app/requirement.txt
WORKDIR /app
RUN pip3 install --upgrade pip && pip3 install -r requirement.txt
WORKDIR /app
COPY . /app/
CMD python3 manage.py makemigrations && python3 manage.py makemigrations && python3 manage.py runserver
```

requirement.txt

```
Django==3.1.3
PyMySQL==1.0.2
```

- **Docker-compose.yml File** (Required for generating Multiple Containers which interact with host Environment)

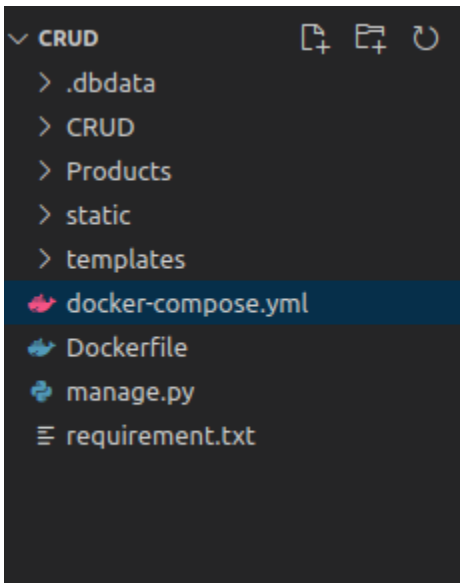
  Downloading Mysql and Mongodb

Defining Ports and Credentials for interaction with django application

docker-compose.yml

```yaml
version: "3.8"
services:
  backend:
    build:
      context: .
      dockerfile: Dockerfile
    ports:
      - 8000:8000
    volumes:
      - .:/app
    depends_on:
      - mongodb
      - db
    image: 'nitesh18400/bdmh_assignment_1'
  db:
    image: mysql:5.7.22
    restart: always
    environment:
      MYSQL_DATABASE: admin
      MYSQL_USER: root
      MYSQL_PASSWORD: root
      MYSQL_ROOT_PASSWORD: root
    volumes:
      - .dbdata:/var/lib/mysql
    ports:
      - 33066:3306
  mongodb:
      container_name: mongodb
      image: mongo
      ports:
        - 27017:27017
      logging:
```

- **Stack File Structure** (for multiple Container)

- 

Product -> Django Application

Template -> Containing HTML Files

Static -> Containing CSS and JS Files

Mange.py -> Main Host file for launching of Website

Requirment.txt -> Containing required python package name with their version