PROJECT NAME : INVENTORY MANGEMENT SYSTEM FOR RETAILERS

ASSIGNMENT – 2

QUESTIONS

1. Create a Flask App

Solution:

Simple Web Application

This is a simple web application using [Python Flask](http://flask.pocoo.org/) and [MySQL](https://www.mysql.com/) database.

This is used in the demonstration of development of Ansible Playbooks.

Below are the steps required to get this working on a base linux system.

- Install all required dependencies
- Install and Configure Web Server
- Start Web Server

1. Install all required dependencies

Python and its dependencies

```
apt-get install -y python python-setuptools python-dev build-essential python-pip python-mysqldb
```

2. Install and Configure Web Server

Install Python Flask dependency

```
pip install flask

pip install flask-mysql
```

- Copy app.py or download it from source repository

- Configure database credentials and parameters

3. Start Web Server

Start web server

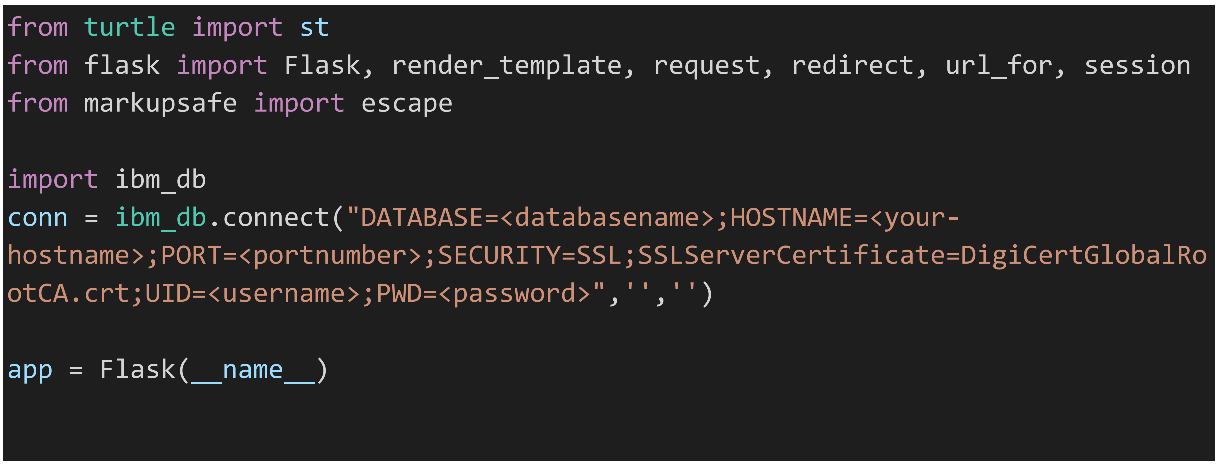
```
FLASK_APP=app.py flask run --host=0.0.0.0
```

4. Test

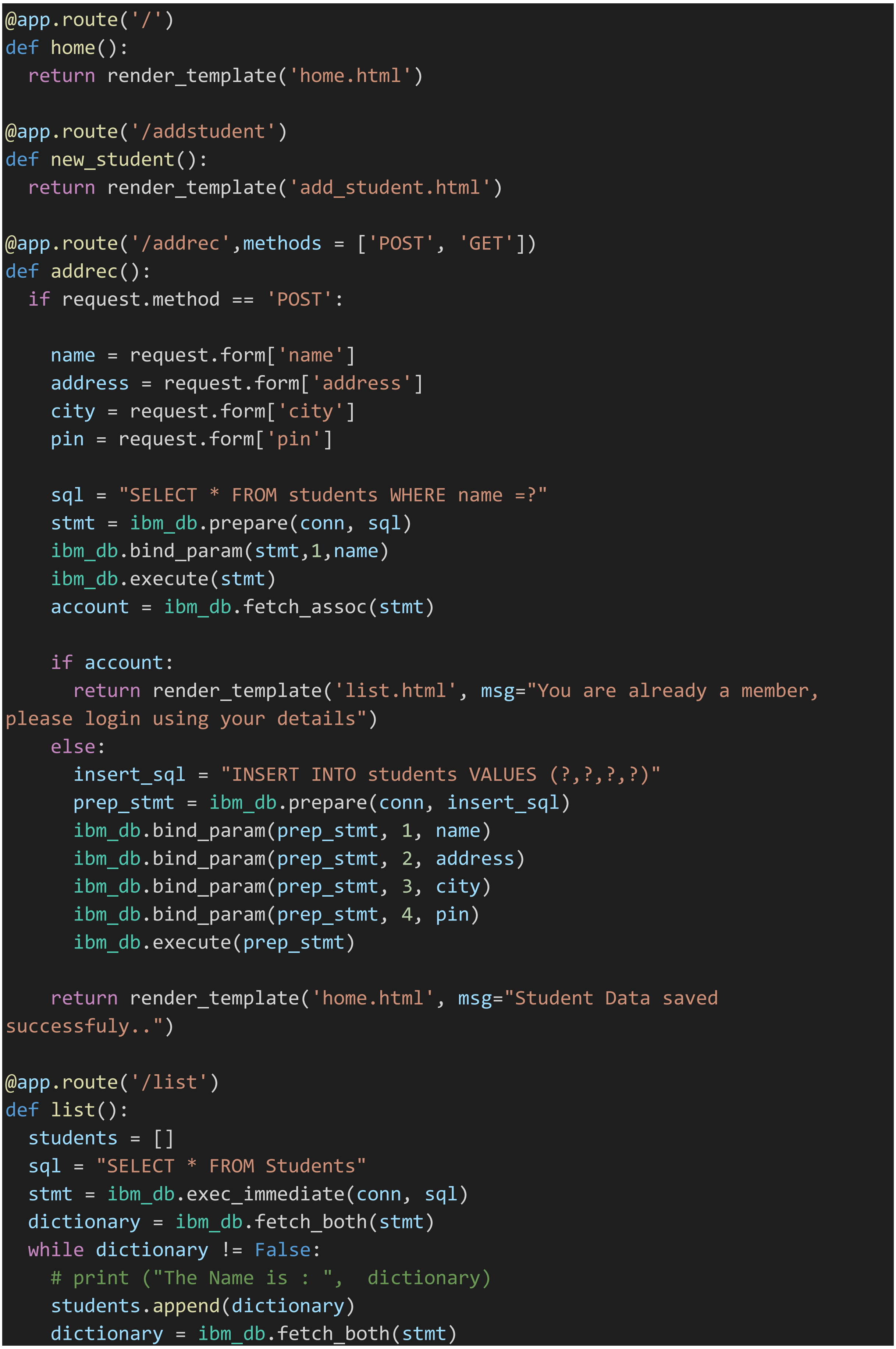
Open a browser and go to URL

```
http://<IP>:5000                    => Welcome

http://<IP>:5000/how%20are%20you        => I am good, how about you?
```

PROGRAM:



```
from turtle import st
from flask import Flask, render_template, request, redirect, url_for, session
from markupsafe import escape

import ibm_db
conn = ibm_db.connect("DATABASE=<databasename>;HOSTNAME=<your-hostname>;PORT=<portnumber>;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=<username>;PWD=<password>",'','')

app = Flask(__name__)
```



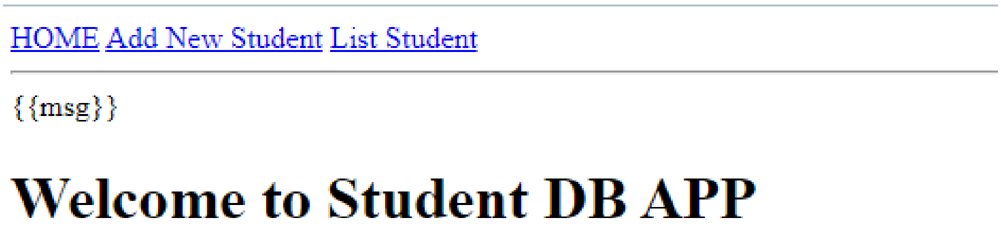
```
@app.route('/')
def home():
  return render_template('home.html')

@app.route('/addstudent')
def new_student():
  return render_template('add_student.html')

@app.route('/addrec',methods = ['POST', 'GET'])
def addrec():
  if request.method == 'POST':

    name = request.form['name']
    address = request.form['address']
    city = request.form['city']
    pin = request.form['pin']

    sql = "SELECT * FROM students WHERE name =?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,name)
    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)

    if account:
      return render_template('list.html', msg="You are already a member,
please login using your details")
    else:
      insert_sql = "INSERT INTO students VALUES (?,?,?,?)"
      prep_stmt = ibm_db.prepare(conn, insert_sql)
      ibm_db.bind_param(prep_stmt, 1, name)
      ibm_db.bind_param(prep_stmt, 2, address)
      ibm_db.bind_param(prep_stmt, 3, city)
      ibm_db.bind_param(prep_stmt, 4, pin)
      ibm_db.execute(prep_stmt)

    return render_template('home.html', msg="Student Data saved
successfuly..")

@app.route('/list')
def list():
  students = []
  sql = "SELECT * FROM Students"
  stmt = ibm_db.exec_immediate(conn, sql)
  dictionary = ibm_db.fetch_both(stmt)
  while dictionary != False:
    # print ("The Name is : ",  dictionary)
    students.append(dictionary)
    dictionary = ibm_db.fetch_both(stmt)
  if students:
    return render_template("list.html", students = students)

@app.route('/delete/<name>')
def delete(name):
  sql = f"SELECT * FROM Students WHERE name='{escape(name)}'"
  print(sql)
  stmt = ibm_db.exec_immediate(conn, sql)
  student = ibm_db.fetch_row(stmt)
  print ("The Name is : ",  student)
  if student:
    sql = f"DELETE FROM Students WHERE name='{escape(name)}'"
    print(sql)
    stmt = ibm_db.exec_immediate(conn, sql)

    students = []
    sql = "SELECT * FROM Students"
    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary != False:
      students.append(dictionary)
      dictionary = ibm_db.fetch_both(stmt)
    if students:
      return render_template("list.html", students = students, msg="Delete
successfully")
```

2. Add the Home page, About Page

Solution:

```
<a href="/">HOME</a>
<a href="/addstudent">Add New Student</a>
<a href="/list">List Student</a>
<hr>

{{msg}}

<h1>Welcome to Student DB APP</h1>
```

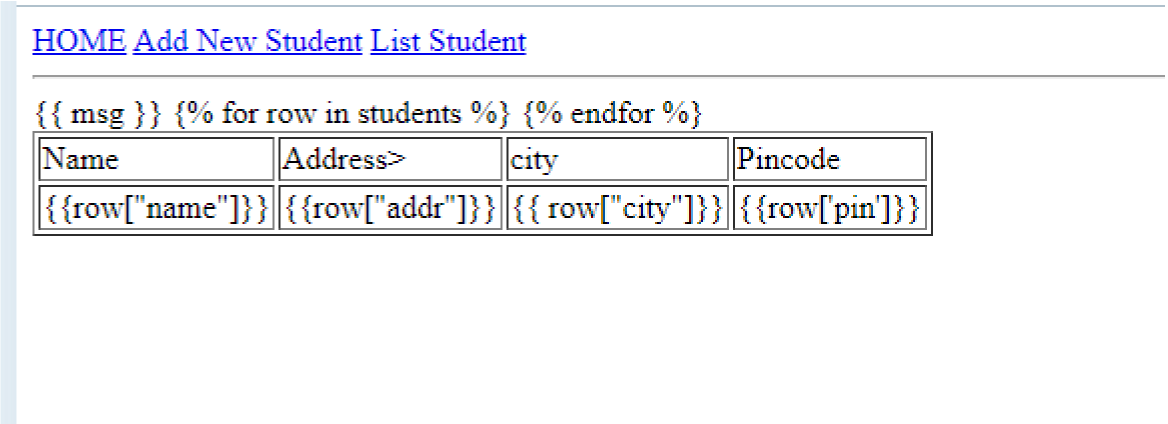
Output:



HOME Add New Student List Student

{{msg}}

Welcome to Student DB APP



HOME Add New Student List Student

{{ msg }} {% for row in students %} {% endfor %}

Name	Address>	city	Pincode
{{row["name"]}}	{{row["addr"]}}	{{ row["city"]}}	{{row['pin']}}

3.Add the Bootstrap

Solution:

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Bootstrap Example</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/
3.4.1/css/bootstrap.min.css">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.
min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstr
ap.min.js"></script>
</head>
<body>

<div class="container-fluid">
<h1>My First Bootstrap Page</h1>
<p>This is some text.</p>
</div>
```

```
</body>
</html>
```

OUTPUT:

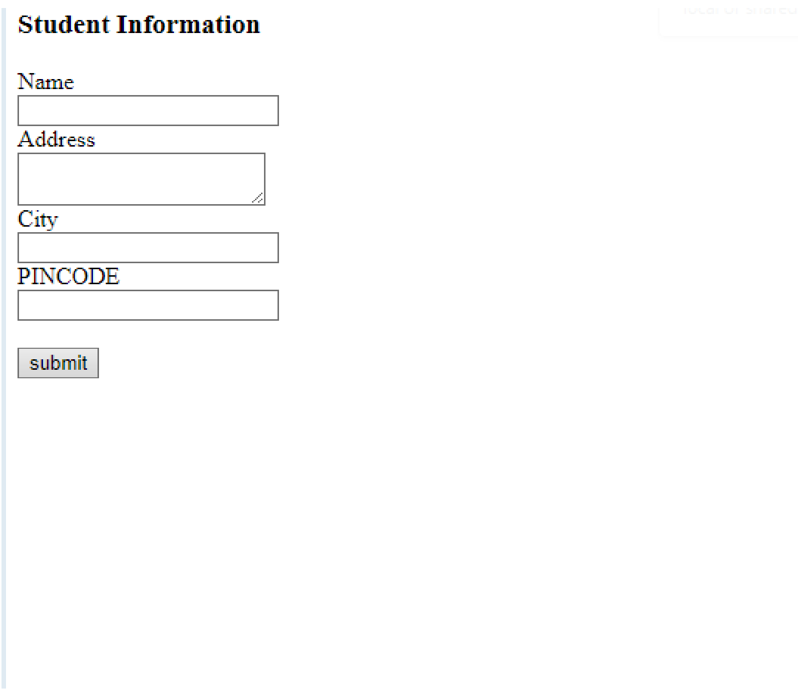
My First Bootstrap Page

4. add the sign in page and add the Signup Page with database connectivity

Solution:

```
<!doctype html>
<html>
   <body>

    <a href="/">HOME</a>
    <a href="/addstudent">Add New Student</a>
    <a href="/list">List Student</a>
    <br><hr>

    {{ msg }}

      <table border = 1>
         <thead>
            <td>Name</td>
            <td>Address</td>
            <td>city</td>
            <td>Pincode</td>
            <td></td>
         </thead>
         {% for row in students %}
            <tr>
               <td>{{row["NAME"]}}</td>
               <td> {{ row["ADDRESS"]}}</td>
               <td>{{row["CITY"]}}</td>
               <td>{{row['PIN']}}</td>
               <td><a href="/delete/{{row['NAME']}}">Delete</a></td>
            </tr>
         {% endfor %}
      </table>


   </body>
</html>
```

OUTPUT:



Student Information

Name

Address

City

PINCODE

5.Use IBM Db2 as Database

Solution:

```
import sqlite3

conn = sqlite3.connect('student_database.db')
print("Opened database successfully")

conn.execute('CREATE TABLE students (name TEXT, addr TEXT, city TEXT, pin
TEXT)')
print("Table created successfully")
conn.close()
```

Output:

{{ msg }}