INVENTORY MANAGEMENT SYSTEM FOR RETAILERS

Domain: Cloud Application Development

Team id: PNT2022TMID10645

Team Members: G.Harini – 4211191021033
P.V.Akshaya - 4211191021009
G.Abirami - 4211191021009
R.Hema - 4211191021037

CREATE FLASK PROJECT

PHYTHON

- **APPLICATION.PY**

```
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=bludb;HOSTNAME=54a2f15b-5c0f-46df-8954-
7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32733;SECURITY=SSL
;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=lkc93724;PWD=zAzNGa6DaNk6xvle",'','')

import smtplib, ssl
## email.mime subclasses
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
## The pandas library is only for generating the current date, which is not necessary for
sending emails
import pandas as pd

app = Flask(__name__)

var_list = []
app.secret_key = 'your secret key'

@app.route('/')
def home():
  if not session.get("name"):
```

```
        return render_template('home.html')
    return render_template('home.html', session = session)

@app.route('/register')
def new_student():
    return render_template('Register.html')



@app.route('/addrec',methods = ['POST', 'GET'])
def addrec():
    if request.method == 'POST':
        fname = request.form['fname']
        lname = request.form['lname']
        cname = request.form['cname']
        state = request.form['state']
        city = request.form['city']
        mobileno = request.form['mobileno']
        emailid = request.form['emailid']
        password = request.form['password']
        pincode = request.form['pincode']

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

        if account:
            users = []
            sql = "SELECT * FROM Users"
            stmt = ibm_db.exec_immediate(conn, sql)
            dictionary = ibm_db.fetch_both(stmt)
            while dictionary != False:
                # print ("The Name is : ",  dictionary)
                users.append(dictionary)
                dictionary = ibm_db.fetch_both(stmt)
```

```
    return render_template('list.html', msg="You are already a member, please login using your details", users = users)
   else:

    var_list.append(fname)
    var_list.append(lname)
    var_list.append(cname)
    var_list.append(state)
    var_list.append(city)
    var_list.append(mobileno)
    var_list.append(emailid)
    var_list.append(password)
    var_list.append(pincode)

    bodytemp = r'D:\IBM\GUIDED PROJECT\INVENTORY MANAGEMENT SYSTEM FOR RETAILERS\templates\email.html'
    with open(bodytemp, "r", encoding='utf-8') as f:
       html= f.read()



    # Set up the email addresses and password. Please replace below with your email address and password
    email_from = 'padhu10a@gmail.com'
    epassword = 'rbjibzkssszsbrjo'
    email_to = emailid

    # Generate today's date to be included in the email Subject
    date_str = pd.Timestamp.today().strftime('%Y-%m-%d')

    # Create a MIMEMultipart class, and set up the From, To, Subject fields
    email_message = MIMEMultipart()
    email_message['From'] = email_from
    email_message['To'] = email_to
    email_message['Subject'] = f'Report email - {date_str}'

    # Attach the html doc defined earlier, as a MIMEText html content type to the MIME message
    email_message.attach(MIMEText(html, "html"))
```

```
    # Convert it as a string
    email_string = email_message.as_string()

    # Connect to the Gmail SMTP server and Send Email
    context = ssl.create_default_context()
    with smtplib.SMTP_SSL("smtp.gmail.com", 465, context=context) as server:
      server.login(email_from, epassword)
      server.sendmail(email_from, email_to, email_string)
    return render_template('notify.html')

@app.route('/confirm')
def confirnation():
  insert_sql = "INSERT INTO Users (FIRSTNAME, LASTNAME, COMPANYNAME, STATE, CITY,
MOBILENO, EMAILID, PASSWORD, PINCODE)  VALUES (?,?,?,?,?,?,?,?,?)"
  prep_stmt = ibm_db.prepare(conn, insert_sql)
  ibm_db.bind_param(prep_stmt, 1, var_list[0])
  ibm_db.bind_param(prep_stmt, 2, var_list[1])
  ibm_db.bind_param(prep_stmt, 3, var_list[2])
  ibm_db.bind_param(prep_stmt, 4, var_list[3])
  ibm_db.bind_param(prep_stmt, 5, var_list[4])
  ibm_db.bind_param(prep_stmt, 6, var_list[5])
  ibm_db.bind_param(prep_stmt, 7, var_list[6])
  ibm_db.bind_param(prep_stmt, 8, var_list[7])
  ibm_db.bind_param(prep_stmt, 9, var_list[8])
  ibm_db.execute(prep_stmt)
  return render_template('confirm.html')



@app.route('/login', methods =['POST', 'GET'])
def login():
   msg = ''
   if request.method == 'POST' and 'email' in request.form and 'password' in request.form:
      email = request.form['email']
      password = request.form['password']
```

```
        sql = "SELECT * FROM Users WHERE EMAILID =? AND PASSWORD =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,email)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)

        if account:
            session['loggedin'] = True
            session['id'] = account['ID']
            session['email'] = account['EMAILID']
            session['name'] = account['FIRSTNAME']
            msg = 'Logged in successfully !'
            return render_template('home.html', msg = msg)
        else:
            msg = 'Incorrect email / password !'
    return render_template('login.html', msg = msg)

@app.route('/logout')
def logout():
    session.pop('loggedin', None)
    session.pop('id', None)
    session.pop('email', None)
    session.pop('name', None)
    return redirect(url_for('home'))

@app.route('/list')
def list():
  users = []
  sql = "SELECT * FROM Users"
  stmt = ibm_db.exec_immediate(conn, sql)
  dictionary = ibm_db.fetch_both(stmt)
  while dictionary != False:
    # print ("The Name is : ",  dictionary)
    users.append(dictionary)
    dictionary = ibm_db.fetch_both(stmt)
```

```
if users:
  return render_template("list.html", users = users , session = session)



return "No users..."
```