Login.html

```
<!{% extends 'base.html' %}
{% block head %}
  <title>Login</title>
{% endblock %}
{% block body %}
<style>
    .divider:after,
    .divider:before {
    content: "";
    flex: 1;
    height: 1px;
    background: #eee;
    }
    .h-custom {
    height: calc(100% - 73px);
    @media (max-width: 450px) {
    .h-custom {
    height: 100%;
    }
    }
    .hlink{
     text-decoration: none;
    </style>
    <section class="vh-100">
     <div class="container-fluid h-custom">
      <div class="row d-flex justify-content-center align-items-center h-100">
        <div class="col-md-9 col-lg-6 col-xl-5">
         <img src="https://imsbucket.s3.jp-tok.cloud-object-
storage.appdomain.cloud/login.png"
          class="img-fluid" alt="Sample image">
        <div class="col-md-8 col-lg-6 col-xl-4 offset-xl-1">
         <form class="mx-1 mx-md-4" action="{{ url_for('login') }}" method="post">
<span style="color: red;">{{ msg }}</span>
          <div class="d-flex align-items-center mb-3 pb-1">
     <span class="h1 fw-bold mb-0">Login</span>
          </div>
          <!-- Email input -->
          <div class="form-outline mb-4">
           <input type="email" name="email" value="" placeholder="Email ID" class="form-
control" />
          </div>
          <!-- Password input -->
          <div class="form-outline mb-3">
           <input type="password" name="password" value="" placeholder="Password"
class="form-control" />
          </diV>
```

```
<div class="text-center text-lg-start mt-4 pt-2">
           <input type="submit" class="btn btn-primary btn-lg" style="width: 125px; display:</pre>
block; margin-left: auto; margin-right: auto;">
       </div>
         Don't have an Account?
          <a href="/register" style="color: rgb(0, 0, 0); text-decoration: none;"><strong
style="font-size: large;">Register</strong></a>
         </form>
       </div>
      </div>
     </div>
  </section>
{% endblock %}
DASHBOARD
Users.html
{% extends 'base.html' %}
{% block head %}
  <title>Register</title>
{% endblock %}
{% block body %}
<style>
</style>
<section class="catogories" >
<div class="container-fluid">
     <div class="row">
       <div class="col-lg-6 col-md-6 col-12 p-1">
               <img src="https://imsbucket.s3.jp-tok.cloud-object-</pre>
storage.appdomain.cloud/register.png" class="img-fluid" style="width: 500px; display:
block; margin-left: auto; margin-right: auto;">
         </div>
       <div class="col-lg-6 col-md-6 col-12 p-1" style="margin-top: 40px;">
        <div class="container register-form">
         <div class="form">
          <div class="d-flex align-items-center mb-3 pb-1">
            <span class="h1 fw-bold mb-0">Register</span>
          </div>
            <div class="form-content">
             <form class="mx-1 mx-md-4" action = "{{ url for('addrec') }}" method =
"POST">
              <div class="row">
                <div class="col-md-6">
                   <div class="flex-row align-items-center mb-4">
                    <input type="text" name="fname" value="" placeholder="First Name"</pre>
class="form-control" />
                   </div>
                  <div class="flex-row align-items-center mb-4">
```

```
<input type="text" name="cname" value="" placeholder="Company</pre>
Name" class="form-control" />
                   </div>
                   <div class="flex-row align-items-center mb-4">
                    <input type="text" name="city" value="" placeholder="City"</pre>
class="form-control" />
                   </div>
                   <div class="flex-row align-items-center mb-4">
                    <input type="text" name="mobileno" value="" placeholder="Mobile
Number" class="form-control" />
                   </div>
                   <div class="flex-row align-items-center mb-4">
                    <input type="password" name="password" value=""</pre>
placeholder="Enter Password" class="form-control" />
                   </div>
                </div>
                <div class="col-md-6">
                   <div class="flex-row align-items-center mb-4">
                    <input type="text" name="lname" value="" placeholder="Last Name"</pre>
class="form-control" />
                   </div>
                   <div class="flex-row align-items-center mb-4">
                    <input type="text" name="state" value="" placeholder="State"</pre>
class="form-control" />
                   </div>
                   <div class="flex-row align-items-center mb-4">
                    <input type="text" name="pincode" value="" placeholder="Pincode"
class="form-control" />
                   </div>
                   <div class="flex-row align-items-center mb-4">
                    <input type="email" name="emailid" value="" placeholder="Email ID"
class="form-control" />
                   </div>
                   <div class="flex-row align-items-center mb-4">
                    <input type="password" name="cpassword" value="" placeholder="Re-
type Password" class="form-control" />
                   </div>
                </div>
              <div class="text-center text-lg-start mt-4 pt-2">
               <input type="submit" class="btn btn-primary btn-lg" style="width: 125px;</pre>
display: block; margin-left: auto; margin-right: auto;">
               Already have an Account?
                <a href="/login" style="color: black; text-decoration: none;"><strong
style="font-size: large;">Login</strong></a>
               </div>
              </form>
            </div>
         </div>
```

```
</div>
  </div>
</section>
{% endblock %}
app.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=zAzNGa6Da
Nk6xvle",",")
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
from datetime import datetime
from flask import Flask
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')
(a) 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\SPRINT 2\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 confirmation():
```

```
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']
     return render template('dashboard/dashboard.html')
    else:
      msg = 'Incorrect email / password!'
  return render template('login.html', msg = msg)
(a)app.route('/dashboard')
def dashboard():
 if session['loggedin'] == True:
   return render template('dashboard/dashboard.html')
 else:
  return redirect(url for('home'))
(a)app.route('/addproduct')
def addproduct():
if session['loggedin'] == True:
   return render template('dashboard/addproduct.html')
 else:
  return redirect(url for('home'))
(a)app.route('/movement')
def movement():
```

```
if session['loggedin'] == True:
  products = []
  sql = "SELECT * FROM Products WHERE HOLDERNAME = ?"
  prep stmt = ibm db.prepare(conn, sql)
  ibm db.bind param(prep stmt, 1, session['name'])
  ibm db.execute(prep stmt)
  dictionary = ibm db.fetch both(prep stmt)
  while dictionary != False:
   # print ("The Name is: ", dictionary)
   products.append(dictionary)
   dictionary = ibm db.fetch both(prep stmt)
if products:
   return render template("dashboard/movement.html", products = products, session =
session)
  else:
   return render template("dashboard/movement.html")
  return redirect(url for('home'))
@app.route('/moveproc',methods = ['POST', 'GET'])
def moveproc():
 if request.method == 'POST':
    pname = request.form['pname']
    quantityout = request.form['quantityout']
    tow = request.form['to']
 insert sql = "UPDATE products SET QUANTITYOUT = ?, TO = ? WHERE
PRODUCTNAME = ? AND HOLDERNAME = ?;"
 prep stmt = ibm db.prepare(conn, insert sql)
 ibm db.bind param(prep stmt, 1, quantityout)
 ibm db.bind param(prep stmt, 2, tow)
 ibm db.bind param(prep stmt, 3, pname)
 ibm db.bind param(prep stmt, 4, session['name'])
 ibm db.execute(prep stmt)
 select sql="SELECT QUANTITYIN from PRODUCTS WHERE PRODUCTNAME = ?
AND HOLDERNAME = ?;"
 prep stmt = ibm db.prepare(conn, select sql)
 ibm db.bind param(prep stmt, 1, pname)
 ibm db.bind param(prep stmt, 2, session['name'])
 ibm db.execute(prep stmt)
 outofstock = ibm db.fetch both(prep stmt)
 if outofstock['QUANTITYIN'] <= int(quantityout):
  bodytemp = r"D:\IBM\GUIDED PROJECT\INVENTORY MANAGEMENT SYSTEM
FOR RETAILERS\SPRINT 4\templates\outofstock.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 = session['email']
   # Generate today's date to be included in the email Subject
```

```
date str = pd.Timestamp.today().strftime('%d-%m-%Y')
   # 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'Warning!!! {pname} - Out Of Stock - {date str}'
   # Attach the html doc defined earlier, as a MIMEText html content type to the MIME
   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)
 products = \Pi
 sql = "SELECT * FROM Products WHERE HOLDERNAME = ?"
 prep stmt = ibm db.prepare(conn, sql)
 ibm db.bind param(prep stmt, 1, session['name'])
 ibm db.execute(prep stmt)
 dictionary = ibm db.fetch both(prep stmt)
 while dictionary != False:
   # print ("The Name is: ", dictionary)
   products.append(dictionary)
   dictionary = ibm db.fetch both(prep stmt)
 return render template('dashboard/movement.html', msg = "Product movement noted!",
products = products)
@app.route('/report')
def report():
 if session['loggedin'] == True:
  products = []
  stockonhand = []
  sql = "SELECT * FROM Products WHERE HOLDERNAME = ?"
  prep stmt = ibm db.prepare(conn, sql)
  ibm db.bind param(prep stmt, 1, session['name'])
  ibm db.execute(prep stmt)
  dictionary = ibm db.fetch both(prep stmt)
  while dictionary != False:
   # print ("The Name is: ", dictionary)
   products.append(dictionary)
   dictionary = ibm db.fetch both(prep stmt)
 for i in products:
    calc = int((i['QUANTITYIN'])) - int(i['QUANTITYOUT'])
    stockonhand.append(str(calc))
  return render template('dashboard/report.html', row row1 =zip(products,stockonhand))
  return redirect(url for('home'))
(a)app.route('/stockupdate')
def stock():
```

```
if session['loggedin'] == True:
  products = []
  sql = "SELECT * FROM Products WHERE HOLDERNAME = ?"
  prep stmt = ibm db.prepare(conn, sql)
  ibm db.bind param(prep stmt, 1, session['name'])
  ibm db.execute(prep stmt)
  dictionary = ibm db.fetch both(prep stmt)
  while dictionary != False:
   # print ("The Name is: ", dictionary)
   products.append(dictionary)
   dictionary = ibm db.fetch both(prep stmt)
  if products:
   return render template("dashboard/stockupdate.html", products = products, session =
session)
  else:
   return render template("dashboard/stockupdate.html")
  return redirect(url for('home'))
@app.route('/proc delete', methods = ['POST', 'GET'])
def proc delete():
      id = request.args.get('pid')
      delete sql = "DELETE FROM products WHERE ID = ? AND HOLDERNAME = ?;"
       prep stmt = ibm db.prepare(conn, delete sql)
       ibm db.bind param(prep stmt, 1, id)
      ibm db.bind param(prep stmt, 2, session['name'])
      ibm db.execute(prep stmt)
      products = []
      sql = "SELECT * FROM Products WHERE HOLDERNAME = ?"
      prep stmt = ibm db.prepare(conn, sql)
      ibm db.bind param(prep stmt, 1, session['name'])
      ibm db.execute(prep stmt)
      dictionary = ibm db.fetch both(prep stmt)
      while dictionary != False:
        # print ("The Name is:", dictionary)
        products.append(dictionary)
        dictionary = ibm db.fetch both(prep stmt)
       return render template('dashboard/stockupdate.html', msg='Product successfully
deleted!', products = products)
@app.route('/proc update', methods = ['POST', 'GET'])
def proc update():
       if request.method == 'POST':
        pname = request.form['pname']
        quantityin = request.form['quantityin']
        pid = request.form['pid']
       update sql = "UPDATE products SET PRODUCTNAME = ?, QUANTITYIN = ?
WHERE ID = ? AND HOLDERNAME = ?;"
       prep stmt = ibm db.prepare(conn, update sql)
       ibm db.bind param(prep stmt, 1, pname)
      ibm db.bind param(prep stmt, 2, quantityin)
      ibm db.bind param(prep stmt, 3, pid)
```

```
ibm db.bind param(prep stmt, 4, session['name'])
      ibm db.execute(prep stmt)
      products = []
      sql = "SELECT * FROM Products WHERE HOLDERNAME = ?"
      prep stmt = ibm db.prepare(conn, sql)
      ibm db.bind param(prep stmt, 1, session['name'])
      ibm db.execute(prep stmt)
      dictionary = ibm db.fetch both(prep stmt)
      while dictionary != False:
        # print ("The Name is: ", dictionary)
       products.append(dictionary)
        dictionary = ibm db.fetch both(prep stmt)
      return render template('dashboard/stockupdate.html', msg='Product successfully
updated!', products = products)
(a)app.route('/addproc',methods = ['POST', 'GET'])
def addproc():
  if request.method == 'POST':
    pname = request.form['pname']
    quantity = request.form['quantity']
    the time = datetime.now()
    the time = the time.replace(second=0, microsecond=0)
    sql = "SELECT * FROM Products WHERE HOLDERNAME =?"
    stmt = ibm db.prepare(conn, sql)
    ibm db.bind param(stmt,1,session['name'])
    ibm db.execute(stmt)
    product = ibm db.fetch assoc(stmt)
    if product:
     if product['PRODUCTNAME']==pname:
      return render template('dashboard/addproduct.html', msg="Product already added!
Add a new product.")
     else:
      sql ="INSERT INTO Products
(PRODUCTNAME, QUANTITYIN, QUANTITYOUT, TO, DATE, HOLDERNAME)
VALUES (?,?,?,?,?);"
      prep stmt = ibm db.prepare(conn, sql)
      ibm db.bind param(prep stmt, 1, pname)
      ibm db.bind param(prep stmt, 2, quantity)
      ibm db.bind param(prep stmt, 3, '0')
      ibm db.bind param(prep stmt, 4, ")
      ibm db.bind param(prep stmt, 5, str(the time))
      ibm db.bind param(prep stmt, 6, session['name'])
      ibm db.execute(prep stmt)
      return render template('dashboard/addproduct.html', msg="Product added")
    else:
      sql ="INSERT INTO Products
(PRODUCTNAME, QUANTITYIN, QUANTITYOUT, TO, DATE, HOLDERNAME)
VALUES (?,?,?,?,?);"
      prep stmt = ibm db.prepare(conn, sql)
      ibm db.bind param(prep stmt, 1, pname)
      ibm db.bind param(prep stmt, 2, quantity)
```

```
ibm db.bind param(prep stmt, 3, '0')
       ibm db.bind param(prep stmt, 4, ")
       ibm db.bind param(prep stmt, 5, str(the time))
       ibm db.bind param(prep stmt, 6, session['name'])
       ibm db.execute(prep stmt)
       return render template('dashboard/addproduct.html', msg="Product added")
(a)app.route('/productlist')
def productlist():
if session['loggedin'] == True:
  products = []
  sql = "SELECT * FROM Products WHERE HOLDERNAME = ?"
  prep stmt = ibm db.prepare(conn, sql)
  ibm db.bind param(prep stmt, 1, session['name'])
  ibm db.execute(prep stmt)
  dictionary = ibm db.fetch both(prep stmt)
  while dictionary != False:
   # print ("The Name is:", dictionary)
   products.append(dictionary)
   dictionary = ibm db.fetch both(prep stmt)
  if products:
   return render template("dashboard/productlist.html", products = products, session =
session)
  else:
   return render template("dashboard/productlist.html")
 else:
  return redirect(url for('home'))
(a)app.route('/contactsupport')
def contactsupport():
if session['loggedin'] == True:
  return render template('dashboard/contactsupport.html')
 else:
  return redirect(url for('home'))
(@app.route('/contactsup', methods = ['POST','GET'])
def contactsup():
if request.method == 'POST':
   name = request.form['name']
   mobileno = request.form['mobileno']
   emailid = request.form['emailid']
   query = request.form['query']
   html = "<h1>Query from, </h1><br/>><b>Name: </b>"+name+"<br/><b>Email ID:
</b>"+emailid+"<br/><b>Contact no: </b>"+mobileno+"<br/><b>Query:
</b><b>"+query+"</b>"
    # 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 = 'imsa3258@gmail.com'
    # 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'Query 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('dashboard/contactsupport.html', msg = "We have mailed your
query to our Support team! Soon they will reach you.")
@app.route('/feedback')
def feedback():
if session['loggedin'] == True:
   return render template('dashboard/feedback.html')
 else:
  return redirect(url for('home'))
(@app.route('/feedbackadd', methods = ['POST','GET'])
def feedbackadd():
if request.method == 'POST':
   interface = request.form['interface']
   availability = request.form['availability']
   userfriendly = request.form['userfriendly']
   chatbot = request.form['chatbot']
   suggest = request.form['suggest']
   sql = "SELECT * FROM Feedback WHERE NAME =?"
   stmt = ibm db.prepare(conn, sql)
   ibm db.bind param(stmt,1,session['name'])
   ibm db.execute(stmt)
   account = ibm db.fetch assoc(stmt)
   if account:
    return render template('dashboard/feedback.html', msg = "Your feedback was submitted
already.")
   else:
    ins sql = "INSERT INTO Feedback
(interface, availability, userfriendly, chatbot, suggest, name) VALUES (?,?,?,?,?);"
    prep stmt = ibm db.prepare(conn, ins sql)
    ibm db.bind param(prep stmt, 1, interface)
    ibm db.bind param(prep stmt, 2, availability)
    ibm db.bind param(prep stmt, 3, userfriendly)
    ibm db.bind param(prep stmt, 4, chatbot)
    ibm db.bind param(prep stmt, 5, suggest)
    ibm db.bind param(prep stmt, 6, session['name'])
    ibm db.execute(prep stmt)
```

```
return render template('dashboard/feedback.html', msg = "Your feedback was
submitted.")
@app.route('/logout')
def logout():
  session['loggedin'] = False
  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..."
mystyle.css
.right
  display: block;
  float: right;
  color: blue;
.bg
  background-image: url('pexels-francesco-ungaro-281260.jpg');
  background-position: center;
  background-repeat: no-repeat;
  background-size: cover;
addproduct.html
<!DOCTYPE html >
  <head>
    <meta charset="utf-8">
    <meta http-equiv="x-ua-compatible" content="ie=edge">
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- Bootstrap Css & Js -->
    <lik href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"</pre>
rel="stylesheet" integrity="sha384-
1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">
```

```
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IIRH9sENBO0LRn5q+8nbTov4+1p"
crossorigin="anonymous"></script>
    <style>
    html,body
       height: 100%;
       margin: 0;
       font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
    </style>
    <!-- CSS here -->
    <link href="static/css/mystyle.css" rel="Stylesheet" />
<body>
  <div style="background-image: url('https://imsbucket.s3.jp-tok.cloud-object-</p>
storage.appdomain.cloud/confirm succ.png');background-position: center; background-
repeat: no-repeat; background-size: contain; background-repeat: no-repeat; height: 100%;">
    <div style="margin-top: 50px;">
    <h1 class="display-6" style="text-align: center;">Your Registration was
successful!</h1>
    <h1 class="display-6" style="text-align: center;">Now, Log In to your account </h1>
    </div>
    <section class="vh-100">
       <div class="container-fluid h-custom">
        <div class="row d-flex justify-content-center align-items-center h-100">
         <div class="col-md-9 col-lg-6 col-xl-5">
         </div>
         <div class="col-md-8 col-lg-6 col-xl-4 offset-xl-1">
            <div class="col-md-6">
            <a href="/login">
              <div class="d-flex justify-content-center mx-4 mb-3 mb-lg-4">
              <button type="button" class="btn btn-primary btn-lg img-</pre>
fluid">Login</button></div></a>
            </div>
         </div>
        </div>
       </div>
      </section>
  </div>
</body>
</html>
addmovement.html
{% extends 'base.html' %}
{% block head %}
  <title>List of Users</title>
{% endblock %}
{% block body %}
```

```
<!-- Hero Area Start-->
     {{ msg }}
     {{ session['name'] }}
        <table border = 1>
           <thead>
               FISRT NAME
               LAST NAME
               COMPANY NAME
               STATE
               CITY
               MOBILENO
               EMAILID
               PASSWORD
                PINCODE
           </thead>
            {% for row in users %}
               {{row["FIRSTNAME"]}}
                   {{ row["LASTNAME"]}}}
                   {\row["COMPANYNAME"]}}
                   {\row['STATE']}} 
                   {\row["CITY"]}}
                   {{ row["MOBILENO"]}}
                   {\row["EMAILID"]}}
                   {\{row['PASSWORD']\}}
                   {{row['PINCODE']}}}
            {% endfor %}
       {% endblock %}
newgroup.html
<!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Transitional //EN"</p>
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<a href="http://www.w3.org/1999/xhtml" xmlns:v="urn:schemas-microsoft-com:vml" xmlns="http://www.w3.org/1999/xhtml" xmlns:v="urn:schemas-microsoft-com:vml" xmlns:v="u
xmlns:o="urn:schemas-microsoft-com:office:office">
<head>
  <!--[if gte mso 9]>
<xml>
  <o:OfficeDocumentSettings>
     <o:AllowPNG/>
     <o:PixelsPerInch>96</o:PixelsPerInch>
  </o:OfficeDocumentSettings>
</xml>
<![endif]-->
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta name="x-apple-disable-message-reformatting">
  <!--[if !mso]><!-->
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```
<!--<![endif]-->
<title></title>
 <style type="text/css">
  @media only screen and (min-width: 620px) {
   .u-row {
    width: 600px !important;
   .u-row .u-col {
    vertical-align: top;
   .u-row .u-col-100 {
    width: 600px !important;
  @media (max-width: 620px) {
   .u-row-container {
    max-width: 100%!important;
    padding-left: 0px !important;
    padding-right: 0px !important;
   .u-row .u-col {
    min-width: 320px !important;
    max-width: 100%!important;
    display: block !important;
   .u-row {
    width: calc(100% - 40px)!important;
   .u-col {
    width: 100%!important;
   .u-col>div {
    margin: 0 auto;
  body {
   margin: 0;
   padding: 0;
  }
  table,
  tr,
  td {
   vertical-align: top;
   border-collapse: collapse;
  p {
   margin: 0;
.ie-container table,
  .mso-container table {
```

```
table-layout: fixed;
   line-height: inherit;
  a[x-apple-data-detectors='true'] {
   color: inherit !important;
   text-decoration: none !important;
  table,
  td {
   color: #000000;
@media (max-width: 480px) {
   #u column 3.v-col-background-color {
    background-color: #3598db !important;
 </style>
 <!--[if !mso]><!-->
 k href="https://fonts.googleapis.com/css?family=Cabin:400,700" rel="stylesheet"
type="text/css">
 <!--<![endif]-->
</head>
<body class="clean-body u body" style="margin: 0;padding: 0;-webkit-text-size-adjust:</p>
100%;background-color: #f9f9f9;color: #000000">
 <!--[if IE]><div class="ie-container"><![endif]-->
 <!--[if mso]><div class="mso-container"><![endif]-->
 <table style="border-collapse: collapse: table-layout: fixed; border-spacing: 0; mso-table-
lspace: 0pt;mso-table-rspace: 0pt;vertical-align: top;min-width: 320px;Margin: 0
auto;background-color: #f9f9f9; width: 100%" cellpadding="0" cellspacing="0">
  <td style="word-break: break-word;border-collapse: collapse !important;vertical-align:
top">
     <!--[if (mso)|(IE)]><table width="100%" cellpadding="0" cellspacing="0"
border="0"><![endif]-->
     <div class="u-row-container" style="padding: 0px;background-color: transparent">
      <div class="u-row" style="Margin: 0 auto;min-width: 320px;max-width:</pre>
600px;overflow-wrap: break-word;word-wrap: break-word;word-break: break-
word;background-color: #ffffff;">
        <div style="border-collapse: collapse; display: table; width: 100%; height:</p>
100%;background-color: transparent;">
         <!--[if (mso)|(IE)]><table width="100%" cellpadding="0" cellspacing="0"
border="0"><td style="padding: 0px;background-color: transparent;"
align="center"><table cellpadding="0" cellspacing="0" border="0"
style="width:600px;"><![endif]-->
         <!--[if (mso)|(IE)]><td align="center" width="600" class="v-col-background-
color" style="width: 600px;padding: 0px;border-top: 0px solid transparent;border-left: 0px
```

```
solid transparent;border-right: 0px solid transparent;border-bottom: 0px solid transparent;"
valign="top"><![endif]-->
        <diy class="u-col u-col-100" style="max-width: 320px:min-width: 600px;display:</p>
table-cell; vertical-align: top;">
          <div class="v-col-background-color" style="height: 100%; width: 100%</p>
!important;">
          <!--[if (!mso)&(!IE)]><!-->
          <div style="height: 100%; padding: 0px;border-top: 0px solid</pre>
transparent;border-left: 0px solid transparent;border-right: 0px solid transparent;border-
bottom: 0px solid transparent;">
            <!--<![endif]-->
            <table style="font-family:'Cabin',sans-serif;" role="presentation"
cellpadding="0" cellspacing="0" width="100%" border="0">
             <td style="overflow-wrap:break-word;word-break:break-
word;padding:0px;font-family:'Cabin',sans-serif;" align="left">
                <img align="right" border="0"</pre>
src="https://assets.unlayer.com/projects/111476/1668447888843-Warning-cuate.png"
alt="Image" title="Image" style="outline: none;text-decoration: none;-ms-interpolation-
mode: bicubic;clear: both;display: inline-block!important;border: none;height: auto;float:
none; width: 100%; max-width: 600px;"
                    width="600"/>
                  <!--[if (!mso)&(!IE)]><!-->
          </div>
          <!--<![endif]-->
         </div>
        </div>
        <!--[if (mso)|(IE)]><![endif]-->
        <!--[if (mso)|(IE)]><![endif]-->
       </div>
      </div>
     </div>
     <div class="u-row-container" style="padding: 0px;background-color: transparent">
      <div class="u-row" style="Margin: 0 auto;min-width: 320px;max-width:</pre>
600px;overflow-wrap: break-word;word-wrap: break-word;word-break: break-
word;background-color: #003399;">
       <div style="border-collapse: collapse; display: table; width: 100%; height:</pre>
100%;background-color: transparent;">
```

```
<!--[if (mso)|(IE)]><table width="100%" cellpadding="0" cellspacing="0"
border="0"><td style="padding: 0px;background-color: transparent;"
align="center"><table cellpadding="0" cellspacing="0" border="0"
style="width:600px;"><![endif]-->
        <!--[if (mso)|(IE)]><td align="center" width="600" class="v-col-background-
color" style="background-color: #3598db;width: 600px;padding: 0px;border-top: 0px solid
transparent;border-left: 0px solid transparent;border-right: 0px solid transparent;border-
bottom: 0px solid transparent;" valign="top"><![endif]-->
        <div id="u column 3" class="u-col u-col-100" style="max-width: 320px;min-</pre>
width: 600px;display: table-cell;vertical-align: top;">
         <div class="v-col-background-color" style="background-color: #3598db;height:</p>
100%; width: 100%! important;">
          <!--[if (!mso)&(!IE)]><!-->
          <div style="height: 100%; padding: 0px;border-top: 0px solid</pre>
transparent;border-left: 0px solid transparent;border-right: 0px solid transparent;border-
bottom: 0px solid transparent;">
           <!--<![endif]-->
           <table style="font-family:'Cabin',sans-serif;" role="presentation"
cellpadding="0" cellspacing="0" width="100%" border="0">
             <td style="overflow-wrap:break-word;word-break:break-
word;padding:40px 10px 10px;font-family:'Cabin',sans-serif;" align="left">
                <img align="center" border="0"
src="https://assets.unlayer.com/projects/111476/1668448081560-vecteezy warning-icon-
png-transparent 9663941 592.png" alt="Image" title="Image" style="outline: none;text-
decoration: none;-ms-interpolation-mode: bicubic;clear: both;display: inline-block
!important;border: none;height: auto;float: none;width: 30%;max-width: 174px;"
                    width="174"/>
                  <table style="font-family: Cabin', sans-serif;" role="presentation"
cellpadding="0" cellspacing="0" width="100%" border="0">
             <td style="overflow-wrap:break-word;word-break:break-
word;padding:10px;font-family:'Cabin',sans-serif;" align="left">
                <div style="color: #e5eaf5; line-height: 140%; text-align: center; word-</pre>
wrap: break-word;">
                 <span style="color:</pre>
#000000; font-size: 20px; line-height: 28px;">WARNING</span>
                </div>
```

```
<table style="font-family:'Cabin',sans-serif;" role="presentation"
cellpadding="0" cellspacing="0" width="100%" border="0">
                           td style="overflow-wrap:break-word;word-break:break-word;padding:0px">td style="overflow-wrap:break-word;word-break:break-word;padding:0px">td style="overflow-wrap:break-word;word-break:break-word;padding:0px">td style="overflow-wrap:break-word;word-break:break-word;padding:0px">td style="overflow-wrap:break-word;padding:0px">td style="overflow-wrap:break-word;padding:0px style="overflow-wrap:break-word;padding:0px;">td style="overflow-wrap:break-word;padding:0px;">td style="overflow-wrap:break-word;padding:0px;">td style="ov
10px 31px;font-family:'Cabin',sans-serif;" align="left">
                                 <div style="color: #e5eaf5; line-height: 140%; text-align: center; word-</pre>
wrap: break-word;">
                                    <span style="font-size:</pre>
24px; line-height: 33.6px;"><strong>You are <span style="color: #000000; font-size: 24px;
line-height: 33.6px;">Out of Stock !</span></strong>
                                      </span>
                                    </div>
                               <!--[if (!mso)&(!IE)]><!-->
                      </div>
                      <!--<![endif]-->
                    </div>
                  </div>
                  <!--[if (mso)|(IE)]><![endif]-->
                  <!--[if (mso)|(IE)]><![endif]-->
               </div>
             </div>
           </div>
           <div class="u-row-container" style="padding: 0px;background-color: transparent">
             <div class="u-row" style="Margin: 0 auto;min-width: 320px;max-width:</pre>
600px;overflow-wrap: break-word;word-wrap: break-word;word-break: break-
word;background-color: #ffffff;">
               <div style="border-collapse: collapse; display: table; width: 100%; height:</pre>
100%;background-color: transparent;">
                  <!--[if (mso)|(IE)]><table width="100%" cellpadding="0" cellspacing="0"
border="0"><td style="padding: 0px;background-color: transparent;"
align="center"><table cellpadding="0" cellspacing="0" border="0"
style="width:600px;"><![endif]-->
                  <!--[if (mso)|(IE)]><td align="center" width="600" class="v-col-background-
color" style="background-color: #000000; width: 600px; padding: 0px; border-top: 0px solid
transparent;border-left: 0px solid transparent;border-right: 0px solid transparent;border-
bottom: 0px solid transparent;" valign="top"><![endif]-->
                  <div class="u-col u-col-100" style="max-width: 320px;min-width: 600px;display:</pre>
table-cell; vertical-align: top;">
                    <div class="v-col-background-color" style="background-color: #000000;height:</pre>
100%; width: 100%! important;">
```

```
<!--[if (!mso)&(!IE)]><!-->
          <div style="height: 100%; padding: 0px;border-top: 0px solid</pre>
transparent;border-left: 0px solid transparent;border-right: 0px solid transparent;border-
bottom: 0px solid transparent;">
           <!--<![endif]-->
           <table style="font-family: Cabin', sans-serif;" role="presentation"
cellpadding="0" cellspacing="0" width="100%" border="0">
            <td style="overflow-wrap:break-word;word-break:break-
word;padding:33px 55px;font-family:'Cabin',sans-serif;" align="left">
               <div style="color: #000000; line-height: 160%; text-align: center; word-</pre>
wrap: break-word;">
                <span style="font-size:</pre>
18px; line-height: 28.8px; color: #ecf0f1;">Please order <span style="color: #3598db; font-
size: 18px; line-height: 28.8px;">new stocks </span>to get rid of the <span style="color:
#3598db; font-size: 18px; line-height: 28.8px;">out-of-stock</span>.</span>
                </div>
              <table style="font-family: Cabin', sans-serif;" role="presentation"
cellpadding="0" cellspacing="0" width="100%" border="0">
            <td style="overflow-wrap:break-word;word-break:break-
word;padding:33px 55px 60px;font-family:'Cabin',sans-serif;" align="left">
               <div style="color: #3598db; line-height: 160%; text-align: center; word-</pre>
wrap: break-word;">
                <span</pre>
style="font-size: 18px; line-height: 28.8px; color: #3598db;">Post queries in the Contact
Support for further clearance!</span>

                <span</pre>
style="font-size: 18px; line-height: 28.8px; color: #3598db;">Thank you!</span>
               </div>
              <!--[if (!mso)&(!IE)]><!-->
          </div>
          <!--<![endif]-->
         </div>
        </div>
        <!--[if (mso)|(IE)]><![endif]-->
        <!--[if (mso)|(IE)]><![endif]-->
       </div>
```

```
</div>
     </div>
     <!--[if (mso)|(IE)]><![endif]-->
    <!--[if mso]></div><![endif]-->
 <!--[if IE]></div><![endif]-->
</body>
</html>
productmovement.html
{% extends 'dashboard/base.html' %}
{% block head %}
  <title>Stock Update</title>
  k href="//maxcdn.bootstrapcdn.com/bootstrap/3.3.0/css/bootstrap.min.css"
rel="stylesheet" id="bootstrap-css">
  link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">
  <script type="text/javascript">
  </script>
  <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-</pre>
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"</p>
integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
crossorigin="anonymous"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"</pre>
integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
crossorigin="anonymous"></script>
  <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></script>
{% endblock %}
{% block body %}
  <div class="home-content">
   <div class="overview-boxes">
   <h1 style="font-size: 40px;" class="fw-bold mb-0">Stock Update</h1>
```

 $< h3 > {\{msg\}} < /h3 >$

<thead>

PRODUCT NAME

```
QUANTITY IN
                15px;">ACTIONS
               </thead>
              {% for row in products %}
               {{row["PRODUCTNAME"]}}
                {{row["QUANTITYIN"]}}}
                <!-- <td><a href="{{ url for('proc update',pid=row['ID']) }}"
onclick="return confirm('Do you want to update {{ row.PRODUCTNAME }}');" style="text-
decoration: none;"><button class="btn btn-info btn-sm" type="submit" style="display: block;
margin-left: auto; margin-right: 1px;">Update</button></a> -->
                <!-- add Modal code-->
                <button type="button" class="btn btn-info" data-toggle="modal" data-</pre>
target="#exampleModal{{row.ID}}" style="display: block; margin-left: auto; margin-right:
1px:">
                  Update</button>
                  <div class="modal fade" id="exampleModal{{row.ID}}" tabindex="-1"</pre>
role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true" style="margin-
top:150px;">
                   <div class="modal-dialog" role="document">
                    <div class="modal-content">
                      <div class="modal-header">
                       <h3 class="modal-title" id="exampleModalLabel" style="font-
weight: bold;">Update Product</h3>
                       <button type="button" class="close" data-dismiss="modal" aria-</pre>
label="Close">
                        <span aria-hidden="true">&times;</span>
                       </button>
                      </div>
                      <div id="modal-cont" class="modal-body" >
                       <form action="{{ url for('proc update') }}" method="POST">
                        <div class="prodname">
                         <input type="hidden" name="pid" value="{{row.ID}}}"/>
                         <label style="font-size: 15px;">Product name</label>
                         <input style="font-size: 15px;" name="pname" type="text"</pre>
class="form-control input-number" value="{{row.PRODUCTNAME}}}" />
                         <label style="font-size: 15px;">Quantity</label>
                         <input style="font-size: 15px;" name="quantityin" type="text"</pre>
class="form-control input-number" />
                        <input type="submit" name="Update" class="btn btn-primary</pre>
btn-lg form-control input-number" style="width: 125px; display: block; margin-top: 50px;
margin-left: auto; margin-right: auto;">
                       </div>
                      </div>
                      <div class="modal-footer">
                       <button type="button" class="btn btn-secondary" data-</pre>
dismiss="modal">Cancel</button>
```

```
< div >
                        </div>
                      </div>
                     </form>
                     </div>
                    </div>
                   </div>
                   <!--end of modal design-->
                 <a href="{{ url for('proc delete',pid=row['ID']) }}" onclick="return"
confirm('Do you want to permanently delete {{ row.PRODUCTNAME }}');">< button
class="btn btn-danger" type="submit">Delete</button></a>
                {% endfor %}
             </div>
            </div>
   </div>
  </div>
 {% endblock %}
newuser.html
{% extends 'dashboard/base.html' %}
{% block head %}
  <title>Stock Update</title>
  link href="//maxcdn.bootstrapcdn.com/bootstrap/3.3.0/css/bootstrap.min.css"
rel="stylesheet" id="bootstrap-css">
  link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">
  <script type="text/javascript">
  </script>
  <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-</pre>
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"</pre>
integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
crossorigin="anonymous"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"</pre>
integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYI"
crossorigin="anonymous"></script>
  <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></script>
{% endblock %}
{% block body %}
```

```
<div class="home-content">
   <div class="overview-boxes">
   <h1 style="font-size: 40px;" class="fw-bold mb-0">Stock Update</h1>
           <br>
           < h3 > {\{msg\}} < /h3 >
            <thead>
              PRODUCT NAME
               QUANTITY IN
               15px;">ACTIONS
              </thead>
             {% for row in products %}
              {{row["PRODUCTNAME"]}}
               {{row["QUANTITYIN"]}}}
               <!-- <td><a href="{{ url for('proc update',pid=row['ID']) }}"
onclick="return confirm('Do you want to update {{ row.PRODUCTNAME }}');" style="text-
decoration: none;"><button class="btn btn-info btn-sm" type="submit" style="display: block;
margin-left: auto; margin-right: 1px;">Update</button></a> -->
               <!-- add Modal code-->
               <button type="button" class="btn btn-info" data-toggle="modal" data-</pre>
target="#exampleModal{{row.ID}}" style="display: block; margin-left: auto; margin-right:
1px;">
                 Update</button>
                 <div class="modal fade" id="exampleModal{{row.ID}}" tabindex="-1"</pre>
role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true" style="margin-
top:150px;">
                  <div class="modal-dialog" role="document">
                   <div class="modal-content">
                    <div class="modal-header">
                     <a>h3 class="modal-title" id="exampleModalLabel" style="font-
weight: bold;">Update Product</h3>
                     <button type="button" class="close" data-dismiss="modal" aria-</pre>
label="Close">
                      <span aria-hidden="true">&times;</span>
                     </button>
                    </div>
                    <div id="modal-cont" class="modal-body" >
                     <form action="{{ url_for('proc update') }}" method="POST">
                      <div class="prodname">
                       <input type="hidden" name="pid" value="{{row.ID}}}"/>
                       <label style="font-size: 15px;">Product name</label>
                       <input style="font-size: 15px;" name="pname" type="text"</pre>
class="form-control input-number" value="{{row.PRODUCTNAME}}}" />
                       <label style="font-size: 15px;">Quantity</label>
```

```
<input style="font-size: 15px;" name="quantityin" type="text"</pre>
class="form-control input-number" />
                          <input type="submit" name="Update" class="btn btn-primary</pre>
btn-lg form-control input-number" style="width: 125px; display: block; margin-top: 50px;
margin-left: auto; margin-right: auto;">
                         </div>
                       </div>
                       <div class="modal-footer">
                         <button type="button" class="btn btn-secondary" data-</pre>
dismiss="modal">Cancel</button>
                         < div >
                         </div>
                       </div>
                      </form>
                      </div>
                     </div>
                   </div>
                    <!--end of modal design-->
                 <a href="{{ url for('proc delete',pid=row['ID']) }}" onclick="return"
confirm('Do you want to permanently delete {{ row.PRODUCTNAME }}');"><button
class="btn btn-danger" type="submit">Delete</button></a>
                {% endfor %}
              </div>
             </div>
            </div>
          </div>
{% endblock %}
Product.html
{% extends 'dashboard/base.html' %}
{% block head %}
  <title>Feedback</title>
  <lik href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"</pre>
rel="stylesheet" integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
crossorigin="anonymous">
{% endblock %}
{% block body %}
<div class="home-content">
       <div class="overview-boxes">
              <section class="catogories" >
                     <div class="container-fluid">
                            <div class="row">
                                   <div class="col-lg-6 col-md-6 col-12 p-1">
                  <img src="static/img/Feedback-cuate.png" class="img-fluid"</pre>
style="width: 500px; display: block; margin-left: auto; margin-right: auto;">
```

```
<div class="col-lg-6 col-md-6 col-12 p-1"</pre>
style="margin-top: 40px;">
                                      <div class="container register-form">
                                     <div class="form">
                 {{msg}}
        <div class="d-flex align-items-center mb-3 pb-1">
                      <span class="h1 fw-bold mb-0">Feedback Form</span>
                                             </div>
                                                   <div class="form-content">
                                                    <form class="" action = "{{
url for('feedbackadd') }}" method = "POST">
                   1) Is our interface good at finding the item present?
              <div class="form-check form-check-inline">
                      <input class="form-check-input" type="radio" name="interface"</pre>
id="inlineRadio1" value="Very satisfied" >
                      <label class="form-check-label" for="inlineRadio1">Very
satisfied</label>
                    </div>
                    <div class="form-check form-check-inline">
                      <input class="form-check-input" type="radio" name="interface"</pre>
id="inlineRadio2" value="satisfied">
                      <label class="form-check-label" for="inlineRadio2">satisfied</label>
                     </div>
                     <div class="form-check form-check-inline">
                      <input class="form-check-input" type="radio" name="interface"</pre>
id="inlineRadio3" value="unsatisfied">
                      <label class="form-check-label"</pre>
for="inlineRadio3">unsatisfied</label>
                    </div>
                    2) Is the availability of stock information accurate?
                   <div class="form-check form-check-inline">
                      <input class="form-check-input" type="radio" name="availability"</pre>
id="inlineRadio1" value="Very satisfied">
                      <label class="form-check-label" for="inlineRadio1">Very
satisfied</label>
                    </div>
                     <div class="form-check form-check-inline">
                      <input class="form-check-input" type="radio" name="availability"</pre>
id="inlineRadio2" value="satisfied">
                     <label class="form-check-label" for="inlineRadio2">satisfied</label>
                     </div>
                     <div class="form-check form-check-inline">
                      <input class="form-check-input" type="radio" name="availability"</pre>
id="inlineRadio3" value="unsatisfied">
                   <label class="form-check-label" for="inlineRadio3">unsatisfied</label>
                    3) Is our service user-friendly?
                     <div class="form-check form-check-inline">
```

```
<input class="form-check-input" type="radio" name="userfriendly"</pre>
id="inlineRadio1" value="Very satisfied" >
                       <label class="form-check-label" for="inlineRadio1">Very
satisfied</label>
                      <div class="form-check form-check-inline">
                       <input class="form-check-input" type="radio" name="userfriendly"</pre>
id="inlineRadio2" value="satisfied">
                       <label class="form-check-label"</pre>
for="inlineRadio2">satisfied</label>
                      <div class="form-check form-check-inline">
                       <input class="form-check-input" type="radio" name="userfriendly"</pre>
id="inlineRadio3" value="unsatisfied">
                       <label class="form-check-label"</pre>
for="inlineRadio3">unsatisfied</label>
                      </div>
                      4) Is our chatbot user-friendly and helpful?
                      <div class="form-check form-check-inline">
                        <input class="form-check-input" type="radio" name="chatbot"</pre>
id="inlineRadio1" value="Very satisfied" >
                        <label class="form-check-label" for="inlineRadio1">Very
satisfied</label>
                       </div>
                       <div class="form-check form-check-inline">
                        <input class="form-check-input" type="radio" name="chatbot"</pre>
id="inlineRadio2" value="satisfied">
                        <label class="form-check-label"</pre>
for="inlineRadio2">satisfied</label>
                       </div>
                       <div class="form-check form-check-inline">
                        <input class="form-check-input" type="radio" name="chatbot"</pre>
id="inlineRadio3" value="unsatisfied">
                        <label class="form-check-label"</pre>
for="inlineRadio3">unsatisfied</label>
                       </div>
                       Is there anything you want share with us?
                       <textarea class="form-control" id="exampleFormControlTextarea1"
rows="3" name="suggest" placeholder="Tell us what you feel"></textarea>
                      <div class="text-center text-lg-start mt-4 pt-2">
                       <input type="submit" class="btn btn-primary btn-lg" style="width:</pre>
125px; display: block; margin-left: auto; margin-right: auto;">
</div>
</form>
                                            </div>
                                            </div>
                                     </div>
                      </div>
              </section>
       </div>
```

```
</div>
{% endblock %}
```

addproduct.html

```
{% extends 'dashboard/base.html' %}
{% block head %}
  <title>Feedback</title>
  <lik href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"</pre>
rel="stylesheet" integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
crossorigin="anonymous">
{% endblock %}
{% block body %}
              <div class="home-content">
       <div class="overview-boxes">
              <section class="catogories" >
                      <div class="container-fluid">
                             <div class="row">
                                    <div class="col-lg-6 col-md-6 col-12 p-1">
<img src="static/img/Feedback-cuate.png" class="img-fluid" style="width: 500px; display:</pre>
block; margin-left: auto; margin-right: auto;"
                                           </div>
                                    <div class="col-lg-6 col-md-6 col-12 p-1"</pre>
style="margin-top: 40px;">
                                     <div class="container register-form">
                                           <div class="form">
                 {{msg}}
                                            <div class="d-flex align-items-center mb-3 pb-
1">
                                           <span class="h1 fw-bold mb-0">Feedback
Form</span>
                                            </div>
       <div class="form-content">
        <form class="" action = "{{ url for('feedbackadd') }}" method = "POST">
                   1) Is our interface good at finding the item present?
              <div class="form-check form-check-inline">
                      <input class="form-check-input" type="radio" name="interface"</pre>
id="inlineRadio1" value="Very satisfied" >
                      <label class="form-check-label" for="inlineRadio1">Very
satisfied</label>
                    </div>
                    <div class="form-check form-check-inline">
                     <input class="form-check-input" type="radio" name="interface"</pre>
id="inlineRadio2" value="satisfied">
                     <label class="form-check-label" for="inlineRadio2">satisfied</label>
                    </div>
                    <div class="form-check form-check-inline">
                     <input class="form-check-input" type="radio" name="interface"</pre>
id="inlineRadio3" value="unsatisfied">
```

```
<label class="form-check-label"</pre>
for="inlineRadio3">unsatisfied</label>
                     </div>
                     2) Is the availability of stock information accurate?
                             <div class="form-check form-check-inline">
                      <input class="form-check-input" type="radio" name="availability"</pre>
id="inlineRadio1" value="Very satisfied">
                      <label class="form-check-label" for="inlineRadio1">Very
satisfied</label>
                     </div>
                     <div class="form-check form-check-inline">
                      <input class="form-check-input" type="radio" name="availability"</pre>
id="inlineRadio2" value="satisfied">
                      <label class="form-check-label" for="inlineRadio2">satisfied</label>
                     </div>
                     <div class="form-check form-check-inline">
                      <input class="form-check-input" type="radio" name="availability"</pre>
id="inlineRadio3" value="unsatisfied">
                      <label class="form-check-label"</pre>
for="inlineRadio3">unsatisfied</label>
                     </div>
                     3) Is our service user-friendly?
                     <div class="form-check form-check-inline">
                       <input class="form-check-input" type="radio" name="userfriendly"</pre>
id="inlineRadio1" value="Very satisfied" >
                       <label class="form-check-label" for="inlineRadio1">Very
satisfied</label>
                      </div>
                      <div class="form-check form-check-inline">
                       <input class="form-check-input" type="radio" name="userfriendly"</pre>
id="inlineRadio2" value="satisfied">
                       <label class="form-check-label"</pre>
for="inlineRadio2">satisfied</label>
                      </div>
                      <div class="form-check form-check-inline">
                       <input class="form-check-input" type="radio" name="userfriendly"</pre>
id="inlineRadio3" value="unsatisfied">
                       <label class="form-check-label"</pre>
for="inlineRadio3">unsatisfied</label>
                      </div>
                      4) Is our chatbot user-friendly and helpful?
                      <div class="form-check form-check-inline">
                        <input class="form-check-input" type="radio" name="chatbot"</pre>
id="inlineRadio1" value="Very satisfied" >
                        <label class="form-check-label" for="inlineRadio1">Very
satisfied</label>
                       <div class="form-check form-check-inline">
                        <input class="form-check-input" type="radio" name="chatbot"</pre>
id="inlineRadio2" value="satisfied">
```

```
<label class="form-check-label"</pre>
for="inlineRadio2">satisfied</label>
                      </div>
                      <div class="form-check form-check-inline">
                        <input class="form-check-input" type="radio" name="chatbot"</pre>
id="inlineRadio3" value="unsatisfied">
                        <label class="form-check-label"</pre>
for="inlineRadio3">unsatisfied</label>
                      </div>
                      Is there anything you want share with us?
                      <textarea class="form-control" id="exampleFormControlTextarea1"
rows="3" name="suggest" placeholder="Tell us what you feel"></textarea>
<div class="text-center text-lg-start mt-4 pt-2">
<input type="submit" class="btn btn-primary btn-lg" style="width: 125px; display: block;</pre>
margin-left: auto; margin-right: auto;">
                                                  </div>
                                                         </form>
                                                  </div>
                                           </div>
                                    </div>
                     </div>
              </section>
       </div>
       </div>
              {% endblock %}
config.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=zAzNGa6Da
Nk6xvle",",")
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
from datetime import datetime
from flask import Flask
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\SPRINT 2\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 confirmation():
 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']
       return render template('dashboard/dashboard.html')
    else:
       msg = 'Incorrect email / password !'
  return render template('login.html', msg = msg)
(a)app.route('/dashboard')
def dashboard():
if session['loggedin'] == True:
  return render template('dashboard/dashboard.html')
 else:
  return redirect(url for('home'))
(a)app.route('/addproduct')
def addproduct():
if session['loggedin'] == True:
   return render template('dashboard/addproduct.html')
 else:
  return redirect(url for('home'))
@app.route('/movement')
def movement():
 if session['loggedin'] == True:
  products = []
  sql = "SELECT * FROM Products WHERE HOLDERNAME = ?"
  prep stmt = ibm db.prepare(conn, sql)
  ibm db.bind param(prep stmt, 1, session['name'])
  ibm db.execute(prep stmt)
  dictionary = ibm db.fetch both(prep stmt)
  while dictionary != False:
   # print ("The Name is:", dictionary)
   products.append(dictionary)
   dictionary = ibm db.fetch both(prep stmt)
  if products:
   return render template("dashboard/movement.html", products = products, session =
session)
  else:
   return render template("dashboard/movement.html")
  return redirect(url for('home'))
@app.route('/moveproc',methods = ['POST', 'GET'])
def moveproc():
 if request.method == 'POST':
    pname = request.form['pname']
    quantityout = request.form['quantityout']
    tow = request.form['to']
```

```
insert sql = "UPDATE products SET QUANTITYOUT = ?, TO = ? WHERE
PRODUCTNAME = ? AND HOLDERNAME = ?;"
 prep stmt = ibm db.prepare(conn, insert sql)
 ibm db.bind param(prep stmt, 1,quantityout)
 ibm db.bind param(prep stmt, 2, tow)
 ibm db.bind param(prep stmt, 3, pname)
 ibm_db.bind_param(prep stmt, 4, session['name'])
 ibm db.execute(prep stmt)
 select sql="SELECT QUANTITYIN from PRODUCTS WHERE PRODUCTNAME = ?
AND HOLDERNAME = ?;"
 prep stmt = ibm db.prepare(conn, select sql)
 ibm db.bind param(prep stmt, 1, pname)
 ibm db.bind param(prep stmt, 2, session['name'])
 ibm db.execute(prep stmt)
 outofstock = ibm db.fetch both(prep stmt)
 if outofstock['QUANTITYIN'] <= int(quantityout):</pre>
   bodytemp = r"D:\IBM\GUIDED PROJECT\INVENTORY MANAGEMENT SYSTEM
FOR RETAILERS\SPRINT 4\templates\outofstock.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 = session['email']
   # Generate today's date to be included in the email Subject
   date str = pd.Timestamp.today().strftime('%d-%m-%Y')
   # 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'Warning!!! {pname} - Out Of Stock - {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)
 products = []
 sql = "SELECT * FROM Products WHERE HOLDERNAME = ?"
 prep stmt = ibm db.prepare(conn, sql)
 ibm db.bind param(prep stmt, 1, session['name'])
 ibm db.execute(prep stmt)
 dictionary = ibm db.fetch both(prep stmt)
 while dictionary != False:
   # print ("The Name is: ", dictionary)
```

```
products.append(dictionary)
   dictionary = ibm db.fetch both(prep stmt)
 return render template('dashboard/movement.html', msg = "Product movement noted!",
products = products)
@app.route('/report')
def report():
 if session['loggedin'] == True:
  products = []
  stockonhand = []
  sql = "SELECT * FROM Products WHERE HOLDERNAME = ?"
  prep stmt = ibm db.prepare(conn, sql)
  ibm db.bind param(prep stmt, 1, session['name'])
  ibm db.execute(prep stmt)
  dictionary = ibm db.fetch both(prep stmt)
  while dictionary != False:
   # print ("The Name is: ", dictionary)
   products.append(dictionary)
   dictionary = ibm db.fetch both(prep stmt)
  for i in products:
    calc = int((i['QUANTITYIN'])) - int(i['QUANTITYOUT'])
    stockonhand.append(str(calc))
  return render template('dashboard/report.html', row row1 =zip(products,stockonhand))
 else:
  return redirect(url for('home'))
@app.route('/stockupdate')
def stock():
 if session['loggedin'] == True:
  products = []
  sql = "SELECT * FROM Products WHERE HOLDERNAME = ?"
  prep stmt = ibm db.prepare(conn, sql)
  ibm db.bind param(prep stmt, 1, session['name'])
  ibm db.execute(prep stmt)
  dictionary = ibm db.fetch both(prep stmt)
  while dictionary != False:
   # print ("The Name is: ", dictionary)
   products.append(dictionary)
   dictionary = ibm db.fetch both(prep stmt)
  if products:
   return render template("dashboard/stockupdate.html", products = products, session =
session)
  else:
   return render template("dashboard/stockupdate.html")
 else:
  return redirect(url for('home'))
@app.route('/proc delete', methods = ['POST', 'GET'])
def proc delete():
       id = request.args.get('pid')
       delete sql = "DELETE FROM products WHERE ID = ? AND HOLDERNAME = ?;"
       prep stmt = ibm db.prepare(conn, delete sql)
       ibm db.bind param(prep stmt, 1, id)
```

```
ibm db.bind param(prep stmt, 2, session['name'])
      ibm db.execute(prep stmt)
      products = []
      sql = "SELECT * FROM Products WHERE HOLDERNAME = ?"
      prep stmt = ibm db.prepare(conn, sql)
      ibm db.bind param(prep stmt, 1, session['name'])
      ibm db.execute(prep stmt)
      dictionary = ibm db.fetch both(prep stmt)
      while dictionary != False:
        # print ("The Name is: ", dictionary)
   products.append(dictionary)
        dictionary = ibm db.fetch both(prep stmt)
      return render template('dashboard/stockupdate.html', msg='Product successfully
deleted!', products = products)
@app.route('/proc update', methods = ['POST', 'GET'])
def proc update():
      if request.method == 'POST':
        pname = request.form['pname']
        quantityin = request.form['quantityin']
        pid = request.form['pid']
      update sql = "UPDATE products SET PRODUCTNAME = ?, QUANTITYIN = ?
WHERE ID = ? AND HOLDERNAME = ?;"
      prep stmt = ibm db.prepare(conn, update sql)
      ibm db.bind param(prep stmt, 1, pname)
      ibm db.bind param(prep stmt, 2, quantityin)
      ibm db.bind param(prep stmt, 3, pid)
      ibm db.bind param(prep stmt, 4, session['name'])
      ibm db.execute(prep stmt)
      products = []
      sql = "SELECT * FROM Products WHERE HOLDERNAME = ?"
      prep stmt = ibm db.prepare(conn, sql)
      ibm db.bind param(prep stmt, 1, session['name'])
      ibm db.execute(prep stmt)
      dictionary = ibm db.fetch both(prep stmt)
      while dictionary != False:
        # print ("The Name is: ", dictionary)
        products.append(dictionary)
        dictionary = ibm db.fetch both(prep stmt)
      return render template('dashboard/stockupdate.html', msg='Product successfully
updated!', products = products)
(@app.route('/addproc',methods = ['POST', 'GET'])
def addproc():
  if request.method == 'POST':
    pname = request.form['pname']
    quantity = request.form['quantity']
    the time = datetime.now()
    the time = the time.replace(second=0, microsecond=0)
    sql = "SELECT * FROM Products WHERE HOLDERNAME =?"
    stmt = ibm db.prepare(conn, sql)
```

```
ibm db.bind param(stmt,1,session['name'])
    ibm db.execute(stmt)
    product = ibm db.fetch assoc(stmt)
    if product:
     if product['PRODUCTNAME']==pname:
      return render template('dashboard/addproduct.html', msg="Product already added!
Add a new product.")
     else:
      sql ="INSERT INTO Products
(PRODUCTNAME, QUANTITYIN, QUANTITYOUT, TO, DATE, HOLDERNAME)
VALUES (?,?,?,?,?);"
      prep stmt = ibm db.prepare(conn, sql)
      ibm db.bind param(prep stmt, 1, pname)
      ibm db.bind param(prep stmt, 2, quantity)
      ibm db.bind param(prep stmt, 3, '0')
      ibm db.bind param(prep stmt, 4, ")
      ibm db.bind param(prep stmt, 5, str(the time))
      ibm db.bind param(prep stmt, 6, session['name'])
      ibm db.execute(prep stmt)
      return render template('dashboard/addproduct.html', msg="Product added")
    else:
      sql ="INSERT INTO Products
(PRODUCTNAME, QUANTITYIN, QUANTITYOUT, TO, DATE, HOLDERNAME)
VALUES (?,?,?,?,?,?);"
      prep stmt = ibm db.prepare(conn, sql)
      ibm db.bind param(prep stmt, 1, pname)
      ibm db.bind param(prep stmt, 2, quantity)
      ibm db.bind param(prep stmt, 3, '0')
      ibm db.bind param(prep stmt, 4, ")
      ibm db.bind param(prep stmt, 5, str(the time))
      ibm db.bind param(prep stmt, 6, session['name'])
      ibm db.execute(prep stmt)
      return render template('dashboard/addproduct.html', msg="Product added")
@app.route('/productlist')
def productlist():
 if session['loggedin'] == True:
  products = []
  sql = "SELECT * FROM Products WHERE HOLDERNAME = ?"
  prep stmt = ibm db.prepare(conn, sql)
  ibm db.bind param(prep stmt, 1, session['name'])
  ibm db.execute(prep stmt)
  dictionary = ibm db.fetch both(prep stmt)
  while dictionary != False:
   # print ("The Name is:", dictionary)
   products.append(dictionary)
   dictionary = ibm db.fetch both(prep stmt)
  if products:
   return render template("dashboard/productlist.html", products = products, session =
session)
  else:
```

```
return render template("dashboard/productlist.html")
 else:
  return redirect(url for('home'))
(a) app.route('/contactsupport')
def contactsupport():
if session['loggedin'] == True:
  return render template('dashboard/contactsupport.html')
 else:
  return redirect(url for('home'))
@app.route('/contactsup', methods = ['POST','GET'])
def contactsup():
if request.method == 'POST':
   name = request.form['name']
   mobileno = request.form['mobileno']
   emailid = request.form['emailid']
   query = request.form['query']
   html = "<h1>Ouerv from, </h1><br/>><b>Name: </b>"+name+"<br/><b>Email ID:
</b>"+emailid+"<br/>br/><b>Contact no: </b>"+mobileno+"<br/>br/><b>Query:
</b><b>"+query+"</b>"
     # 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 = 'imsa3258@gmail.com'
    # 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'Query 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('dashboard/contactsupport.html', msg = "We have mailed your
query to our Support team! Soon they will reach you.")
@app.route('/feedback')
def feedback():
if session['loggedin'] == True:
   return render template('dashboard/feedback.html')
  return redirect(url for('home'))
@app.route('/feedbackadd', methods = ['POST','GET'])
```

```
def feedbackadd():
if request.method == 'POST':
   interface = request.form['interface']
   availability = request.form['availability']
   userfriendly = request.form['userfriendly']
   chatbot = request.form['chatbot']
   suggest = request.form['suggest']
   sql = "SELECT * FROM Feedback WHERE NAME =?"
   stmt = ibm db.prepare(conn, sql)
   ibm db.bind param(stmt,1,session['name'])
   ibm db.execute(stmt)
   account = ibm db.fetch assoc(stmt)
   if account:
    return render template('dashboard/feedback.html', msg = "Your feedback was submitted
already.")
   else:
    ins sql = "INSERT INTO Feedback
(interface, availability, userfriendly, chatbot, suggest, name) VALUES (?,?,?,?,?);"
    prep stmt = ibm db.prepare(conn, ins sql)
    ibm db.bind param(prep stmt, 1, interface)
    ibm db.bind param(prep stmt, 2, availability)
    ibm db.bind param(prep stmt, 3, userfriendly)
    ibm db.bind param(prep stmt, 4, chatbot)
    ibm db.bind param(prep stmt, 5, suggest)
    ibm db.bind param(prep stmt, 6, session['name'])
    ibm db.execute(prep stmt)
return render template('dashboard/feedback.html', msg = "Your feedback was submitted.")
@app.route('/logout')
def logout():
  session['loggedin'] = False
  session.pop('id', None)
  session.pop('email', None)
  session.pop('name', None)
  return redirect(url for('home'))
(a)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..."
```

Products.html

```
<!DOCTYPE html >
  <head>
     <meta charset="utf-8">
    <meta http-equiv="x-ua-compatible" content="ie=edge">
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- Bootstrap Css & Js -->
    k href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">
     <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IIRH9sENBO0LRn5q+8nbTov4+1p"
crossorigin="anonymous"></script>
    <style>
    html,body
       height: 100%;
       margin: 0;
       font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
    </style>
    <!-- CSS here -->
    <link href="static/css/mystyle.css" rel="Stylesheet" />
<body>
  <div style="background-image: url('static/img/vecteezy 3d-online-airline-ticket-buying-</p>
confirm-concept-icon-with-3d 10916024 536.png');background-position: center;
background-repeat: no-repeat; background-size: contain; background-repeat: no-repeat;
height: 100%;">
    <div style="margin-top: 50px;">
     <h1 class="display-6" style="text-align: center;">Your Registration was
successful!</h1>
    <h1 class="display-6" style="text-align: center;">Now, Log In to your account </h1>
    </div>
     <section class="vh-100">
       <div class="container-fluid h-custom">
        <div class="row d-flex justify-content-center align-items-center h-100">
         <div class="col-md-9 col-lg-6 col-xl-5">
         </div>
         <div class="col-md-8 col-lg-6 col-xl-4 offset-xl-1">
           <div class="col-md-6">
            <a href="/login">
              <div class="d-flex justify-content-center mx-4 mb-3 mb-lg-4">
              <button type="button" class="btn btn-primary btn-lg img-</pre>
fluid">Login</button></div></a>
           </div>
         </div>
        </div>
```

```
</div>
</section>
</div>
</body>
</html>
```