## Login.html

```
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-</pre>
objectstorage.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"
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=""
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"</pre>
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=""</pre>
placeholder="Retype Password" class="form-control" />
                   </div>
                 </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=98538591-7217-4024-b027-
8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud;PORT=30875;SECURI
TY
=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=cxz16087;PWD=0ZMLpfUq
nDBTiHzz
",",") 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)
   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
                 email_from = 'padhu10a@gmail.com'
and password
                                                         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_message['To'] = email_to
= email_from
   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
          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'] =
                           session['name'] =
account['EMAILID']
account['FIRSTNAME']
                             return
render_template('dashboard/dashboard.html')
                                                else:
      msg = 'Incorrect email / password !'
return render_template('login.html', msg = msg)
@app.route('/dashboard')
def dashboard():
 if session['loggedin'] == True:
  return render_template('dashboard/dashboard.html')
else:
  return
redirect(url for('home'))
@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")
else:
  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 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()
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 =
П
 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'])
                                      return render_template('dashboard/report.html',
    stockonhand.append(str(calc))
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)
                              dictionary =
products.append(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,
                        ibm_db.execute(prep_stmt)
4, session['name'])
      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)
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:
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,
                      ibm_db.bind_param(prep_stmt, 6, session['name'])
5, str(the_time))
ibm_db.execute(prep_stmt)
                                return
render template('dashboard/addproduct.html', msg="Product added")
@app.route('/productlist') def
productlist():
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 =
                               while dictionary != False:
ibm_db.fetch_both(prep_stmt)
   # 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'))
@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 =
                        html = "<h1>Query from, </h1><br/><b>Name:
request.form['query']
</b>"+name+"<br/><b>Email ID:
</b>"+emailid+"<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 from = 'ragu53@gmail.com'
email address and password
                                                                     epassword =
'rbjibzkssszsbrjo'
                    email_to = 'vignesh63@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)
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-
                   background-size: cover;
repeat: no-repeat;
}
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 -->
    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('https://s3.jp-tok.cloud-</pre>
object storage.appdomain.cloud//vigneshramdass/assign%203.pdf');background-position:
center; backgroundrepeat: 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
imgfluid">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: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,
        vertical-
td {
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>
<br/><body class="clean-body u_body" style="margin: 0;padding: 0;-webkit-text-size-adjust:
100%; background-color: #f9f9f9; color: #000000">
 <!--[if IE]><div class="ie-container"><![endif]-->
 <!--[if mso]><div class="mso-container"><![endif]-->
 tablelspace: Opt;mso-table-rspace: Opt;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:
breakword;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-
backgroundcolor" style="width: 600px;padding: 0px;border-top: 0px solid
transparent;border-left: Opx solid transparent;border-right: Opx solid transparent;border-
bottom: 0px solid transparent;" valign="top"><![endif]-->
        <div class="u-col u-col-100" style="max-width: 320px;min-width:</pre>
600px;display: table-cell;vertical-align: top;">
          <div class="v-col-background-color" style="height: 100%; width: 100%</pre>
!important;">
           <!--[if (!mso)&(!IE)]><!-->
```

```
<div style="height: 100%; padding: 0px;border-top: 0px solid</pre>
transparent;border-left: Opx solid transparent;border-right: Opx solid
transparent; borderbottom: 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:breakword;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-
interpolationmode: 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: 600px;overflow-</p>
wrap: break-word; word-wrap: break-word; word-break: breakword; 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-
backgroundcolor" style="background-color: #3598db;width: 600px;padding: 0px;border-
top: Opx solid transparent;border-left: Opx solid transparent;border-right: Opx solid
transparent; borderbottom: Opx solid transparent; "valign="top"><![endif]-->
        <div id="u_column_3" class="u-col u-col-100" style="max-width:</pre>
320px;minwidth: 600px;display: table-cell;vertical-align: top;">
          <div class="v-col-background-color" style="background-color: #3598db;height:</pre>
100%; width: 100%! important;">
           <!--[if (!mso)&(!IE)]><!-->
           <div style="height: 100%; padding: 0px;border-top: 0px solid</pre>
transparent;border-left: Opx solid transparent;border-right: Opx solid
transparent;borderbottom: 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:breakword;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-
iconpng-transparent_9663941_592.png" alt="Image" title="Image" style="outline:
none;textdecoration: 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:breakword;padding:10px;font-family:'Cabin',sans-serif;" align="left">
                <div style="color: #e5eaf5; line-height: 140%; text-align: center;</pre>
wordwrap: 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
10px 31px;font-family:'Cabin',sans-serif;" align="left">
                <div style="color: #e5eaf5; line-height: 140%; text-align: center;</pre>
wordwrap: 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: 600px;overflow-</p>
wrap: break-word; word-wrap: break-word; word-break: breakword; 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-
backgroundcolor" style="background-color: #000000; width: 600px; padding: 0px; border-
top: Opx solid transparent;border-left: Opx solid transparent;border-right: Opx solid
transparent; borderbottom: Opx solid transparent; "valign="top"><![endif]-->
         <div class="u-col u-col-100" style="max-width: 320px;min-width:</pre>
600px;display:
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: Opx solid transparent;border-right: Opx solid
transparent; borderbottom: 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:breakword;padding:33px 55px;font-family:'Cabin',sans-serif;" align="left">
                 <div style="color: #000000; line-height: 160%; text-align: center;</pre>
wordwrap: break-word;">
                  <span style="font-size:</pre>
18px; line-height: 28.8px; color: #ecf0f1;">Please order <span style="color: #3598db;
fontsize: 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:breakword;padding:33px 55px 60px;font-family:'Cabin',sans-serif;" align="left">
                 <div style="color: #3598db; line-height: 160%; text-align: center;</pre>
wordwrap: break-word;">
                  center;"><span style="font-size: 18px; line-height: 28.8px; color: #3598db;">Post queries
in the Contact
```

```
Support for further clearance!</span>

<span style="font-size:</pre>
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/GpGFF93hXpG5Kk
N" 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+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>
           <hr>>
           < 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="textdecoration: 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"
datatarget="#exampleModal{{row.ID}}" style="display: block; margin-left: auto; margin-
right:
```

Update</button>

1px;">

```
<div class="modal fade" id="exampleModal{{row.ID}}" tabindex="-</pre>
       role="dialog"
                          aria-labelledby="exampleModalLabel"
                                                                     aria-hidden="true"
style="margintop:150px;">
                     <div class="modal-dialog" role="document">
                      <div class="modal-content">
                       <div class="modal-header">
                        <h3 class="modal-title" id="exampleModalLabel"
style="fontweight: bold;">Update Product</h3>
                        <button type="button" class="close" data-dismiss="modal"
arialabel="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}}}" />
                           <br>
                           <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"
datadismiss="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>
  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/GpGFF93hXpG5Kk
N" 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+76PVCmY1"
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
                            <th style="font-size:
15px;">QUANTITY IN
                                          <th colspan="2"
style="text-align: center; font-size:
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="textdecoration: 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"
datatarget="#exampleModal{{row.ID}}" style="display: block; margin-left: auto; margin-
right:
1px;">
                  Update</button>
                  <div class="modal fade" id="exampleModal{{row.ID}}" tabindex="-</pre>
1"
       role="dialog"
                        aria-labelledby="exampleModalLabel"
                                                                aria-hidden="true"
style="margintop:150px;">
                   <div class="modal-dialog" role="document">
                    <div class="modal-content">
                      <div class="modal-header">
                       <h3 class="modal-title" id="exampleModalLabel"
style="fontweight: bold;">Update Product</h3>
                       <button type="button" class="close" data-dismiss="modal"
arialabel="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
btn-lg form-control input-number" style="width: 125px; display: block; margin-top: 50px;
margin-left: auto; margin-right: auto;">
                      </div>
                      <div class="modal-footer">
                       <button type="button" class="btn btn-secondary"
datadismiss="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>
  k href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASj
C" 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;</pre>
display: 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"</pre>
for="inlineRadio2">satisfied</label>
                     <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"</pre>
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"
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>
                       <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>
  k href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
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" style="margin-top: 40px;">
                                    <div class="container register-form">
                                          <div class="form">
```

```
{{msg}}
                                               <div class="d-flex align-items-center mb-3</pre>
                                              <span class="h1 fw-bold mb-0">Feedback
pb1">
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"</pre>
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"</pre>
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>
                       <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 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;">
```

## config.py

request.form['cname']

```
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=98538591-7217-4024-b027-
8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud;PORT=30875;SECURI
TY
=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=cxz16087;PWD=0ZMLpfUq
nDBTiHzz\\
",",") 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(\underline{\quad}name\underline{\quad})
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 ==
          fname = request.form['fname']
lname = request.form['lname']
```

```
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)
                                         users = []
                           if account:
   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)
   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
                                               email_message['From']
fields
         email_message = MIMEMultipart()
= 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',
                               if request.method == 'POST' and 'email' in
'GET']) def login():
                     msg = "
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)
@app.route('/dashboard') def
dashboard():
                           if
session['loggedin'] == True:
  return render_template('dashboard/dashboard.html')
else:
  return
redirect(url_for('home'))
```

```
@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)
   return render_template("dashboard/movement.html")
else:
  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"))
message
   # 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 =
                               while dictionary != False:
ibm_db.fetch_both(prep_stmt)
   # 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.")
      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'))
@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
                                                             mobileno =
request.method == 'POST':
                              name = request.form['name']
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/>br/><b>Query:
</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
         email_message = MIMEMultipart()
                                               email_message['From'] =
fields
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()
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,
              ibm db.bind param(prep stmt, 2, availability)
interface)
ibm_db.bind_param(prep_stmt, 3, userfriendly)
                                                   ibm_db.bind_param(prep_stmt, 4,
             ibm db.bind param(prep stmt, 5, suggest)
chatbot)
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..."
```

## **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-</pre>
buyingconfirm-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
imgfluid">Login</button></div></a>
            </div>
         </div>
        </div>
       </div>
      </section>
  </div>
</body>
</html>
```