Sprint-3

Team ID	PNT2022TMID52447
Project Name	Smart Fashion Recommender System

Source code:

```
from flask import Flask, render_template, flash, request,session
from flask import Flask, render_template, request, jsonify
import datetime
import re



import ibm_db
import pandas
import ibm_db_dbi
from sqlalchemy import create_engine

engine = create_engine('sqlite://',
                       echo = False)

dsn_hostname = "fbd88901-ebdb-4a4f-a32e-
9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud"
dsn_uid = "pnv79770"
dsn_pwd = "4C8CDWDfkqreIAYX"
dsn_driver = "{IBM DB2 ODBC DRIVER}"
dsn_database = "BLUDB"
dsn_port = "32731"
dsn_protocol = "TCPIP"
dsn_security = "SSL"

dsn = (
    "DRIVER={0};"
    "DATABASE={1};"
    "HOSTNAME={2};"
    "PORT={3};"
    "PROTOCOL={4};"
    "UID={5};"
    "PWD={6};"
    "SECURITY={7};").format(dsn_driver, dsn_database, dsn_hostname, dsn_port,
dsn_protocol, dsn_uid, dsn_pwd,dsn_security)
```

```
try:
    conn = ibm_db.connect(dsn, "", "")
    print ("Connected to database: ", dsn_database, "as user: ", dsn_uid, "on host:
", dsn_hostname)

except:
    print ("Unable to connect: ", ibm_db.conn_errormsg() )


app = Flask(__name__)
app.config.from_object(__name__)
app.config['SECRET_KEY'] = '7d441f27d441f27567d441f2b6176a'


@app.route("/")
def homepage():

    return render_template('index.html')


@app.route("/AdminLogin")
def AdminLogin():

    return render_template('AdminLogin.html')

@app.route("/NewUser")
def NewUser():

    return render_template('NewUser.html')
@app.route("/UserLogin")
def UserLogin():

    return render_template('UserLogin.html')









@app.route("/AdminHome")
def AdminHome():

    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)

    selectQuery = "SELECT * from regtb "
    dataframe = pandas.read_sql(selectQuery, pd_conn)

    dataframe.to_sql('Employee_Data',
                     con=engine,
                     if_exists='append')
```

```
    # run a sql query
    data = engine.execute("SELECT * FROM Employee_Data").fetchall()

    return render_template('AdminHome.html', data=data)


@app.route("/NewProduct")
def NewProduct():

    return render_template('NewProduct.html')

@app.route("/ProductInfo")
def ProductInfo():
    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)

    selectQuery = "SELECT * from protb "
    dataframe = pandas.read_sql(selectQuery, pd_conn)

    dataframe.to_sql('Employee_Data',
                     con=engine,
                     if_exists='append')

    # run a sql query
    print(engine.execute("SELECT * FROM Employee_Data").fetchall())

    return render_template('ProductInfo.html', data=engine.execute("SELECT * FROM
Employee_Data").fetchall())



@app.route("/SalesInfo")
def SalesInfo():

    return render_template('SalesInfo.html')




@app.route("/Search")
def Search():

    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)

    selectQuery = "SELECT * from protb "
    dataframe = pandas.read_sql(selectQuery, pd_conn)

    dataframe.to_sql('Employee_Data',
                     con=engine,
                     if_exists='append')

    # run a sql query
    print(engine.execute("SELECT * FROM Employee_Data").fetchall())
```

```
    return render_template('ViewProduct.html', data=engine.execute("SELECT * FROM
Employee_Data").fetchall())



@app.route("/viewproduct", methods=['GET', 'POST'])
def viewproduct():

    searc = request.form['subcat']




    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)

    selectQuery =  "SELECT * from protb where SubCategory like '%" + searc + "%' "
    dataframe = pandas.read_sql(selectQuery, pd_conn)

    dataframe.to_sql('Employee_Data',
                     con=engine,
                     if_exists='append')

    # run a sql query
    print(engine.execute("SELECT * FROM Employee_Data").fetchall())

    return render_template('ViewProduct.html', data=engine.execute("SELECT * FROM
Employee_Data").fetchall())




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

        name1 = request.form['name']
        gender1 = request.form['gender']
        Age = request.form['age']
        email = request.form['email']
        address = request.form['address']
        pnumber = request.form['phone']
        uname = request.form['uname']
        password = request.form['psw']

        conn = ibm_db.connect(dsn, "", "")

        insertQuery = "INSERT INTO regtb VALUES ('" + name1 + "','" + gender1 + "','"
+ Age + "','" + email + "','" + pnumber + "','" + address + "','" + uname + "','" +
password + "')"
        insert_table = ibm_db.exec_immediate (conn, insertQuery)
        print(insert_table)
```

```
    return render_template('userlogin.html')

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


        file = request.files['fileupload']
        file.save("static/upload/" + file.filename)


        ProductId =request.form['pid']
        Gender =request.form['gender']
        Category =request.form['cat']
        SubCategory=request.form['subcat']
        ProductType=request.form['ptype']
        Colour=request.form['color']
        Usage=request.form['usage']
        ProductTitle=request.form['ptitle']
        price = request.form['price']
        Image= file.filename
        ImageURL="static/upload/" + file.filename



        conn = ibm_db.connect(dsn, "", "")

        insertQuery = "INSERT INTO protb VALUES ('"+ ProductId +"','" + Gender +
"','" + Category + "','" + SubCategory + "','" + ProductType + "','" + Colour +
"','"+Usage +"','"+ProductTitle+"','"+ Image +"','"+ ImageURL +"','"+ price +"')"
        insert_table = ibm_db.exec_immediate(conn, insertQuery)

        data1 = 'Record Saved!'
        return render_template('goback.html', data=data1)







@app.route("/userlogin", methods=['GET', 'POST'])
def userlogin():
    error = None
    if request.method == 'POST':
        username = request.form['uname']
        password = request.form['password']
        session['uname'] = request.form['uname']


        conn = ibm_db.connect(dsn, "", "")
```

```
        pd_conn = ibm_db_dbi.Connection(conn)

        selectQuery = "SELECT * from regtb where UserName='" + username + "' and
password='" + password + "'"
        dataframe = pandas.read_sql(selectQuery, pd_conn)

        if dataframe.empty:
            data1 = 'Username or Password is wrong'
            return render_template('goback.html', data=data1)
        else:
            print("Login")
            selectQuery = "SELECT * from regtb where UserName='" + username + "' and
password='" + password + "'"
            dataframe = pandas.read_sql(selectQuery, pd_conn)



            dataframe.to_sql('Employee_Data',
                       con=engine,
                       if_exists='append')

            # run a sql query
            print(engine.execute("SELECT * FROM Employee_Data").fetchall())

            return render_template('UserHome.html', data=engine.execute("SELECT *
FROM Employee_Data").fetchall())




@app.route("/adminlogin", methods=['GET', 'POST'])
def adminlogin():
    error = None
    if request.method == 'POST':
        username = request.form['uname']
        password = request.form['password']
        conn = ibm_db.connect(dsn, "", "")
        pd_conn = ibm_db_dbi.Connection(conn)


        if(username=="admin" and password=="admin"):
            selectQuery = "SELECT * from regtb "
            dataframe = pandas.read_sql(selectQuery, pd_conn)

            dataframe.to_sql('Employee_Data', con=engine, if_exists='append')

            # run a sql query
            print(engine.execute("SELECT * FROM Employee_Data").fetchall())

            return render_template('AdminHome.html', data=engine.execute("SELECT *
FROM Employee_Data").fetchall())
        else:
            data1 = 'Username or Password is wrong'
            return render_template('goback.html', data=data1)
```

```
@app.route("/Remove", methods=['GET'])
def Remove():


    pid = request.args.get('id')
    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)

    insertQuery = "Delete from protb  where id='"+ pid +"'"
    insert_table = ibm_db.exec_immediate(conn, insertQuery)


    selectQuery = "SELECT * from protb "
    dataframe = pandas.read_sql(selectQuery, pd_conn)

    dataframe.to_sql('Employee_Data',
                     con=engine,
                     if_exists='append')

    # run a sql query
    print(engine.execute("SELECT * FROM Employee_Data").fetchall())

    return render_template('ProductInfo.html', data=engine.execute("SELECT * FROM
Employee_Data").fetchall())



@app.route("/fullInfo")
def fullInfo():
    pid = request.args.get('pid')
    session['pid'] = pid


    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)

    selectQuery = "SELECT * FROM protb where ProductId='" + pid + "' "
    dataframe = pandas.read_sql(selectQuery, pd_conn)

    dataframe.to_sql('Employee_Data',
                     con=engine,
                     if_exists='append')

    # run a sql query
    print(engine.execute("SELECT * FROM Employee_Data").fetchall())



    return render_template('ProductFullInfo.html', data=engine.execute("SELECT * FROM
```

```
Employee_Data").fetchall())



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


        uname = session['uname']
        pid = session['pid']

        qty = request.form['qty']

        ctype = request.form['ctype']
        cardno = request.form['cardno']
        cvno = request.form['cvno']


        Bookingid = ''
        ProductName =''
        UserName= uname
        Mobile=''
        Email=''
        Qty = qty
        Amount=''


        CardType = ctype
        CardNo = cardno
        CvNo = cvno
        date = datetime.datetime.now().strftime('%d-%b-%Y')


        conn = ibm_db.connect(dsn, "", "")
        pd_conn = ibm_db_dbi.Connection(conn)
        selectQuery = "SELECT * FROM protb where ProductId='" + pid + "' "
        dataframe = pandas.read_sql(selectQuery, pd_conn)

        dataframe.to_sql('Employee_Data',con=engine,if_exists='append')
        data = engine.execute("SELECT * FROM Employee_Data").fetchall()
        for item in data:
            ProductName = item[8]
            price = item[11]
            print(price)
            Amount = float(price) * float(Qty)

            print(Amount)

        selectQuery1 ="SELECT  *  FROM  regtb where  UserName='" + uname + "'"
        dataframe = pandas.read_sql(selectQuery1, pd_conn)

        dataframe.to_sql('regtb', con=engine, if_exists='append')
        data1 = engine.execute("SELECT * FROM regtb").fetchall()
```

```
        for item1 in data1:
            Mobile = item1[5]
            Email = item1[4]




        selectQuery = "SELECT  *  FROM  booktb"
        dataframe = pandas.read_sql(selectQuery, pd_conn)

        dataframe.to_sql('booktb', con=engine, if_exists='append')
        data2 = engine.execute("SELECT * FROM booktb").fetchall()
        count = 0

        for item in data2:
            count+=1




        Bookingid="BOOKID00" + str(count)


        insertQuery = "INSERT INTO booktb VALUES ('" + Bookingid + "','"+ ProductName
+"','" + price + "','" + uname + "','" + Mobile + "','" + Email + "','" + str(Qty) +
"','" + str(Amount) + "','"+ str(CardType) +"','"+ str(CardNo) +"','"+ str(CvNo)
+"','"+ str(date) +"')"
        insert_table = ibm_db.exec_immediate(conn, insertQuery)

        sendmsg(Email,"order received delivery in one week ")







        selectQuery = "SELECT * FROM booktb where  UserName= '" + uname + "' "
        dataframe = pandas.read_sql(selectQuery, pd_conn)

        dataframe.to_sql('booktb1', con=engine, if_exists='append')
        data = engine.execute("SELECT * FROM booktb1").fetchall()

        return render_template('UOrderInfo.html', data=data)



def sendmsg(Mailid,message):
    import smtplib
    from email.mime.multipart import MIMEMultipart
    from email.mime.text import MIMEText
    from email.mime.base import MIMEBase
    from email import encoders

    fromaddr = "sampletest685@gmail.com"
```

```
    toaddr = Mailid

    # instance of MIMEMultipart
    msg = MIMEMultipart()

    # storing the senders email address
    msg['From'] = fromaddr

    # storing the receivers email address
    msg['To'] = toaddr

    # storing the subject
    msg['Subject'] = "Alert"

    # string to store the body of the mail
    body = message

    # attach the body with the msg instance
    msg.attach(MIMEText(body, 'plain'))

    # creates SMTP session
    s = smtplib.SMTP('smtp.gmail.com', 587)

    # start TLS for security
    s.starttls()

    # Authentication
    s.login(fromaddr, "hneucvnontsuwgpj")

    # Converts the Multipart msg into a string
    text = msg.as_string()

    # sending the mail
    s.sendmail(fromaddr, toaddr, text)

    # terminating the session
    s.quit()


@app.route("/UOrderInfo")
def UOrderInfo():

    uname = session['uname']

    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * FROM booktb where  UserName= '" + uname + "' "
    dataframe = pandas.read_sql(selectQuery, pd_conn)
    dataframe.to_sql('booktb1', con=engine, if_exists='append')
    data = engine.execute("SELECT * FROM booktb1").fetchall()


    return render_template('UOrderInfo.html', data=data)
```

```
@app.route("/UserHome")
def UserHome():

    uname = session['uname']


    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * FROM regtb where  UserName= '" + uname + "' "
    dataframe = pandas.read_sql(selectQuery, pd_conn)
    dataframe.to_sql('booktb1', con=engine, if_exists='append')
    data = engine.execute("SELECT * FROM booktb1").fetchall()


    return render_template('UserHome.html', data=data)


@app.route("/ASalesInfo")
def ASalesInfo():
    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * FROM booktb  "
    dataframe = pandas.read_sql(selectQuery, pd_conn)
    dataframe.to_sql('booktb', con=engine, if_exists='append')
    data = engine.execute("SELECT * FROM booktb").fetchall()

    return render_template('ASalesInfo.html', data=data)

def main():
    app.run(debug=True, use_reloader=True)

if __name__ == '__main__':
    main()
```