```
from flask import
Flask,render_template,request,redirect,session,make_response,url_for
import sqlite3 as sql
from functools import wraps
import datetime
import re
from datetime import timedelta


import ibm_db
app=Flask(__name__)
app.secret_key = 'jackiechan'


hostname = "815fa4db-dc03-4c70-869a-
a9cc13f33084.bs2io90l08kqb1od8lcg.databases.appdomain.cloud"
uid = "tkx67682"
pwd = "o7CLe1RKTEB89jC8"
driver = "{IBM DB2 ODBC DRIVER}"
db = "bludb"
port = "30367"
protocol = "TCPIP"
cert = "DigiCertGlobalRootCA.crt"

dsn = (
    "DATABASE={0};"
    "HOSTNAME={1};"
    "PORT={2};"
    "UID={3};"
    "SECURITY=SSL;"
    "SSLServerCertificate={4};"
    "PWD={5};"
).format(db, hostname, port, uid, cert, pwd)

print(dsn)

conn = ibm_db.connect(dsn, "", "")
email = 'harish19gmail.com'

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

print(account)
@app.route('/')
def root():
    return render_template("login.html")

@app.route('/signup', methods=['POST', 'GET'])
def signup():
    mg = ''
    if request.method == "POST":
        username = request.form['username']
        email = request.form['email']
        pw = request.form['password']
        sql = 'SELECT * FROM USERS WHERE email =?'
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, email)
        ibm_db.execute(stmt)
        acnt = ibm_db.fetch_assoc(stmt)
        print(acnt)

        if acnt:
            mg = 'Account already exits!!'
        elif not re.match(r'[^@]+@[^@]+\.[^@]+', email):
            mg = 'Please enter the avalid email address'
        elif not re.match(r'[A-Za-z0-9]+', username):
            mg = 'name must contain only character and number'
        else:
            insert_sql = 'INSERT INTO USERS (USERNAME,EMAIL,PASSWORD)
VALUES (?,?,?)'
            pstmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(pstmt, 1, username)
            # ibm_db.bind_param(pstmt,4,"123456789")
            ibm_db.bind_param(pstmt, 2, email)
            ibm_db.bind_param(pstmt, 3, pw)
            print(pstmt)
            ibm_db.execute(pstmt)
            mg = 'You have successfully registered click login!'
            return render_template("login.html", meg=mg)

    return render_template("signup.html", meg=mg)

def rewrite(url):
    view_func, view_args = app.create_url_adapter(request).match(url)
    return app.view_functions[view_func](**view_args)


def login_required(f):
    @wraps(f)
    def decorated_function(*args, **kwargs):
        if "id" not in session:
            return redirect(url_for('login'))
        return f(*args, **kwargs)
    return decorated_function


@app.route('/dashboard', methods=['POST', 'GET'])
@login_required
def dashBoard():
    sql = "SELECT * FROM STOCKS"
    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_assoc(stmt)
    stocks = []

    while dictionary != False:
        stocks.append(dictionary)
        print(f"The ID is : ", dictionary["NAME"])
        print(f"The name is : ", dictionary["QUANTITY"])
        dictionary = ibm_db.fetch_assoc(stmt)

    return render_template("dashboard.html", data=stocks)


@app.route('/orders', methods=['POST', 'GET'])
@login_required
def orders():
    query = "SELECT * FROM orders"
    stmt = ibm_db.exec_immediate(conn, query)
    dictionary = ibm_db.fetch_assoc(stmt)
    orders = []
    while dictionary != False:
        orders.append(dictionary)
        dictionary = ibm_db.fetch_assoc(stmt)
    return render_template("orders.html", data=orders)


@app.route('/suppliers', methods=['POST', 'GET'])
@login_required
def suppliers():
    sql = "SELECT * FROM suppliers"
    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_assoc(stmt)
    suppliers = []
    orders_assigned = []
    while dictionary != False:
        suppliers.append(dictionary)
        orders_assigned.append(dictionary['NAME'])
        dictionary = ibm_db.fetch_assoc(stmt)

# get order ids from orders table and identify unassigned order ids
    sql = "SELECT STOCKS_ID FROM orders"
    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_assoc(stmt)
    order_ids = []
    while dictionary != False:
        order_ids.append(dictionary['STOCKS_ID'])
        dictionary = ibm_db.fetch_assoc(stmt)

    unassigned_order_ids = set(order_ids) - set(orders_assigned)
    return
render_template("suppliers.html",data=suppliers,order_ids=unassigned_order_
ids)


@app.route('/profile', methods=['POST', 'GET'])
@login_required
def profile():
    if request.method == "GET":
        try:
            email = session['id']
            insert_sql = 'SELECT * FROM users WHERE EMAIL=?'
            pstmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(pstmt, 1, email)
            ibm_db.execute(pstmt)
            dictionary = ibm_db.fetch_assoc(pstmt)
            print(dictionary)
        except Exception as e:
            msg = e
        finally:
            # print(msg)
            return render_template("profile.html", data=dictionary)


@app.route('/logout', methods=['GET'])
@login_required
def logout():
    print(request)
    resp = make_response(render_template("login.html"))
    session.clear()
    return resp
```

```
 <!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.c
ss" rel="stylesheet"
    integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
crossorigin="anonymous" />
  <link rel="stylesheet" href="static/css/style.css" />
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.
min.js"
    integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
    crossorigin="anonymous"></script>

  <!-- <link rel="stylesheet" href="{{url_for('static',
filename='css/style.css')}}"> -->
  {% block head%}{% endblock %}
</head>

<body>
  <div class="container mt-5">{% block body %} {% endblock %}</div>
</body>

</html>
```