```
from __future__ import print_function

from audioop import add

import datetime

from unicodedata import name

import sib_api_v3_sdk

from sib_api_v3_sdk.rest import ApiException

from pprint import pprint

from flask import Flask, render_template, request, redirect, url_for, session, flash

from markupsafe import escape

from flask import *

import ibm_db

import datetime

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=1bbf73c5-d84a-4bb0-85b9-
ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud;PORT=32286;SECURITY=SSL;SSL
ServerCertificate=;UID=zdb94008;PWD=9CbgTMAJjAF0Vqob", '', '')

print(conn)

print("connection successful...")


#app = Flask(__name__)

app.secret_key = ''

app = Flask(__name__, template_folder = 'templates')

app.config['SECRET_KEY'] = ''

app.config['MAIL_SERVER'] = 'smtp.sendgrid.net'

app.config['MAIL_PORT'] = 587

app.config['MAIL_USE_TLS'] = True

app.config['MAIL_USERNAME'] = 'apikey'

app.config['MAIL_PASSWORD'] = ''

app.config['MAIL_DEFAULT_SENDER'] = 'manjupriyak2002@gmail.com'


@app.route('/')

def home():

   message = "TEAM ID : PNT2022TMID29808" +" "+ "BATCH ID : B1-1M3E "
```

```
    return render_template('index.html',mes=message)


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


    return render_template('index.html')

@app.route('/siginpage', methods=['POST','GET'])
def siginpage():
    return render_template('siginpage.html')


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

    return render_template('signinpageagent.html')




@app.route('/signuppage', methods=['POST', 'GET'])
def signuppage():
    return render_template('signuppage.html')

@app.route('/agentRegister', methods=['POST', 'GET'])
def agentRegister():
    return render_template('agentregister.html')
```

```
@app.route('/forgotpass', methods=['POST', 'GET'])
def forgotpass():
    return render_template('forgot.hinttml')



@app.route('/newissue/<name>', methods=['POST', 'GET'])
def newissue(name):
    name = name
    return render_template('complaint.html',msg=name)



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

    try:
        global randomnumber
        ida = request.form['custid']
        print(ida)
        global id
        id = ida
        sql = "SELECT EMAIL,NAME FROM Customer WHERE id=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, ida)
        ibm_db.execute(stmt)
        emailf = ibm_db.fetch_both(stmt)
        while emailf != False:
```

```
            e = emailf[0]
            n = emailf[1]
            break

        configuration = sib_api_v3_sdk.Configuration()
        configuration.api_key['api-key'] = ''

        api_instance = sib_api_v3_sdk.TransactionalEmailsApi(
            sib_api_v3_sdk.ApiClient(configuration))
        subject = "Verification for Password"
        html_content = "<html><body><h1>Your verification Code is : <h2>" + \
            str(randomnumber)+"</h2> </h1> </body></html>"
        sender = {"name": "IBM CUSTOMER CARE REGISTRY",
                  "email": "manjupriyak2002@gmail.com"}
        to = [{"email": e, "name": n}]
        reply_to = {"email": "manjupriyak2002@gmail.com", "name": "IBM"}
        headers = {"Some-Custom-Name": "unique-id-1234"}
        params = {"parameter": "My param value",
                  "subject": "Email Verification"}
        send_smtp_email = sib_api_v3_sdk.SendSmtpEmail(
            to=to, reply_to=reply_to, headers=headers, html_content=html_content, params=params,
sender=sender, subject=subject)

        api_response = api_instance.send_transac_email(send_smtp_email)

        pprint(api_response)
        message = "Email send to:"+e+" for password"
        flash(message, "success")

    except ApiException as e:
        print("Exception when calling SMTPApi->send_transac_email: %s\n" % e)
```

```
        flash("Error in sending mail")
    except:
        flash("Your didn't Signin with this account")
    finally:
        return render_template('forgot.html')

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

    try:
        global randomnumber
        ida = request.form['custid']
        print(ida)
        global id
        id = ida
        sql = "SELECT EMAIL,NAME FROM AGENT WHERE id=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, ida)
        ibm_db.execute(stmt)
        emailf = ibm_db.fetch_both(stmt)
        while emailf != False:
            e = emailf[0]
            n = emailf[1]
            break

        configuration = sib_api_v3_sdk.Configuration()
        configuration.api_key['api-key'] = ""

        api_instance = sib_api_v3_sdk.TransactionalEmailsApi(
            sib_api_v3_sdk.ApiClient(configuration))
        subject = "Verification for Password"
```

```
        html_content = "<html><body><h1>Your verification Code is : <h2>" + \
            str(randomnumber)+"</h2> </h1> </body></html>"
        sender = {"name": "IBM CUSTOMER CARE REGISTRY",
                  "email": "manjupriyak2002@gmail.com"}
        to = [{"email": e, "name": n}]
        reply_to = {"email": "manjupriyak2002@gmail,com", "name": "IBM"}
        headers = {"Some-Custom-Name": "unique-id-1234"}
        params = {"parameter": "My param value",
                  "subject": "Email Verification"}
        send_smtp_email = sib_api_v3_sdk.SendSmtpEmail(
            to=to, reply_to=reply_to, headers=headers, html_content=html_content, params=params,
sender=sender, subject=subject)


        api_response = api_instance.send_transac_email(send_smtp_email)


        pprint(api_response)
        message = "Email send to:"+e+" for OTP"
        flash(message, "success")


    except ApiException as e:
        print("Exception when calling SMTPApi->send_transac_email: %s\n" % e)
        flash("Error in sending mail")
    except:
        flash("Your didn't Signin with this account")
    finally:
        return render_template('forgot.html')
```

```
@app.route('/admin', methods=['POST', 'GET'])
def admin():
    userdatabase = []
    sql = "SELECT * FROM customer"
    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary != False:
        userdatabase.append(dictionary)
        dictionary = ibm_db.fetch_both(stmt)
    if userdatabase:
        sql = "SELECT COUNT(*) FROM customer;"
        stmt = ibm_db.exec_immediate(conn, sql)
        user = ibm_db.fetch_both(stmt)

    users = []
    sql = "select * from ISSUE"
    stmt = ibm_db.exec_immediate(conn, sql)
    dict = ibm_db.fetch_both(stmt)
    while dict != False:
        users.append(dict)
        dict = ibm_db.fetch_both(stmt)
    if users:
        sql = "SELECT COUNT(*) FROM ISSUE;"
        stmt = ibm_db.exec_immediate(conn, sql)
        count = ibm_db.fetch_both(stmt)

    agent = []
    sql = "SELECT * FROM AGENT"
    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary != False:
```

```
        agent.append(dictionary)
        dictionary = ibm_db.fetch_both(stmt)

    if agent:
        sql = "SELECT COUNT(*) FROM AGENT;"
        stmt = ibm_db.exec_immediate(conn, sql)
        cot = ibm_db.fetch_both(stmt)

    return
render_template("admin.html",complaint=users,users=userdatabase,agents=agent,message=user[0]
,issue=count[0],msgagent = cot[0])



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

    otp = request.form['otpv']
    if otp == 'C':
        try:
            insert_sql = f"delete from customer"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.execute(prep_stmt)
            flash("delected successfully the Customer", "success")
        except:
            flash("No data found in Customer", "danger")
        finally:
            return redirect(url_for('signuppage'))
    if otp == 'A':
        try:
            insert_sql = f"delete from AGENT"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.execute(prep_stmt)
```

```
            flash("delected successfully the Agents", "success")
        except:
            flash("No data found in Agents", "danger")
        finally:
           return redirect(url_for('signuppage'))

    if otp == 'C':
        try:
            insert_sql = f"delete from AGENT"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.execute(prep_stmt)
            flash("delected successfully the Complaints", "success")
        except:
            flash("No data found in Complaints", "danger")
        finally:
            return redirect(url_for('signuppage'))

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

            id = request.form['idn']
            global hello
            hello = id
            password = request.form['password']
            print(id, password)
            if id == '1111' and password == '1111':
                return redirect(url_for('admin'))

            sql = f"select * from customer where id='{escape(id)}' and password='{escape(password)}'"
```

```
            stmt = ibm_db.exec_immediate(conn, sql)
            data = ibm_db.fetch_both(stmt)

            if data:
                session["name"] = escape(id)
                session["password"] = escape(password)
                return redirect(url_for("welcome"))

            else:
                flash("Mismatch in credetials", "danger")
        except:
            flash("Error in Insertion operation", "danger")

    return render_template('signinpage.html')


@app.route('/welcome', methods=['POST', 'GET'])
def welcome():
    try:
        id = hello
        sql = "SELECT ID,DATE,TOPIC,SERVICE_TYPE,SERVICE_AGENT,DESCRIPTION,STATUS FROM ISSUE
WHERE CUSTOMER_ID =?"
        agent = []
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, id)
        ibm_db.execute(stmt)
        otpf = ibm_db.fetch_both(stmt)
        while otpf != False:
            agent.append(otpf)
            otpf = ibm_db.fetch_both(stmt)

        sql = "SELECT COUNT(*) FROM ISSUE WHERE CUSTOMER_ID = ?"
```

```
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, id)
        ibm_db.execute(stmt)
        t = ibm_db.fetch_both(stmt)

        return render_template("welcome.html",agent=agent,message=t[0])
    except:
        return render_template("welcome.html")

@app.route('/loginagent', methods=['GET', 'POST'])
def loginagent():
    if request.method == 'POST':
        try:
            global loginagent
            id = request.form['idn']
            loginagent = id
            password = request.form['password']

            sql = f"select * from AGENT where id='{escape(id)}' and password='{escape(password)}'"
            stmt = ibm_db.exec_immediate(conn, sql)
            data = ibm_db.fetch_both(stmt)

            if data:
                session["name"] = escape(id)
                session["password"] = escape(password)
                return redirect(url_for("agentwelcome"))

            else:
                flash("Mismatch in credetials", "danger")
        except:
            flash("Error in Insertion operation", "danger")
```

```
    return render_template("signinpageagent.html")


@app.route('/delete/<ID>')
def delete(ID):
    sql = f"select * from customer where Id='{escape(ID)}'"
    print(sql)
    stmt = ibm_db.exec_immediate(conn, sql)
    student = ibm_db.fetch_row(stmt)
    if student:
        sql = f"delete from customer where id='{escape(ID)}'"
        stmt = ibm_db.exec_immediate(conn, sql)

        flash("Delected Successfully", "success")
        return redirect(url_for("admin"))




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

        try:
            x = datetime.datetime.now()
            y = x.strftime("%Y-%m-%d %H:%M:%S")
            name1 = request.form['name']
            email = request.form['email']
            password = request.form['password']
            phonenumber = request.form['phonenumber']
            service = request.form['service']
            address = request.form['address']
```

```
        city = request.form['city']
        state = request.form['state']
        country = request.form['country']
        link = request.form['link']

        sql = "SELECT * FROM AGENT 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)

        if account:
            flash("Record Aldready found", "success")
        else:
            print("exec")
            insert_sql = "INSERT INTO AGENT
(NAME,EMAIL,PASSWORD,PHONENUMBER,SERVICE_AGENT,ADDRESS,CITY,STATE,COUNTRY,RESUM
E_LINK,DATE) VALUES(?,?,?,?,?,?,?,?,?,?,?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt, 1, name1)
            ibm_db.bind_param(prep_stmt, 2, email)
            ibm_db.bind_param(prep_stmt, 3, password)
            ibm_db.bind_param(prep_stmt, 4, phonenumber)
            ibm_db.bind_param(prep_stmt, 5, service)
            ibm_db.bind_param(prep_stmt, 6, address)
            ibm_db.bind_param(prep_stmt, 7, city)
            ibm_db.bind_param(prep_stmt, 8, state)
            ibm_db.bind_param(prep_stmt, 9, country)
            ibm_db.bind_param(prep_stmt, 10, link)
            ibm_db.bind_param(prep_stmt, 11, y)

            ibm_db.execute(prep_stmt)
```

```
            flash("Record stored Successfully", "success")
            sql = "SELECT ID FROM AGENT WHERE email=?"
            stmt = ibm_db.prepare(conn, sql)
            ibm_db.bind_param(stmt, 1, email)
            ibm_db.execute(stmt)
            hi = ibm_db.fetch_tuple(stmt)

            configuration = sib_api_v3_sdk.Configuration()
            configuration.api_key['api-key'] =""

            api_instance = sib_api_v3_sdk.TransactionalEmailsApi(
            sib_api_v3_sdk.ApiClient(configuration))
            subject = "Registering Account in Customer Care Registry"
            html_content = " <html><body><h1>Thanks for Registering into Customer Care
Registry</h1> <h2>Your Account Id is :"+str(hi[0])+"</h2><h2>With Regards:</h2><h3>Customer
Care Registry</h3> </body></html>"
            sender = {"name": "IBM CUSTOMER CARE REGISTRY",
              "email": "manjupriyak2002@gmail.com"}
            to = [{"email": email, "name": name1}]
            reply_to = {"email": "manjupriyak2002@gmail.com", "name": "IBM"}
            headers = {"Some-Custom-Name": "unique-id-1234"}
            params = {"parameter": "My param value",
              "subject": "Email Verification"}
            send_smtp_email = sib_api_v3_sdk.SendSmtpEmail(
            to=to, reply_to=reply_to, headers=headers, html_content=html_content, params=params,
sender=sender, subject=subject)

            api_response = api_instance.send_transac_email(send_smtp_email)

            pprint(api_response)

        except:
```

```
            flash("Error in Insertion Operation", "danger")
        finally:
            return redirect(url_for("agentRegister"))
            con.close()

    return render_template('agentregister.html')



@app.route('/completed/<DESCRIPTION>', methods=['GET', 'POST'])
def completed(DESCRIPTION):
    status ="Completed"
    try:

        sql = "UPDATE ISSUE SET STATUS = ? WHERE DESCRIPTION =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,status)
        ibm_db.bind_param(stmt,2,DESCRIPTION)
        ibm_db.execute(stmt)

        flash("Successful","success")
        return redirect(url_for('agentwelcome'))
    except:
        flash("No record found","danger")
        return redirect(url_for('agentwelcome'))

@app.route('/deletecomplaint/<ID>')
def deletecomplaint(ID):
    sql = f"select * from ISSUE where ID='{escape(ID)}'"
    print(sql)
    stmt = ibm_db.exec_immediate(conn, sql)
    student = ibm_db.fetch_row(stmt)
```

```
    if student:
        sql = f"delete from ISSUE where ID='{escape(ID)}'"
        stmt = ibm_db.exec_immediate(conn, sql)
        users = []
        flash("Delected Successfully", "success")

        return redirect(url_for("admin"))



if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8080, debug=True)
```