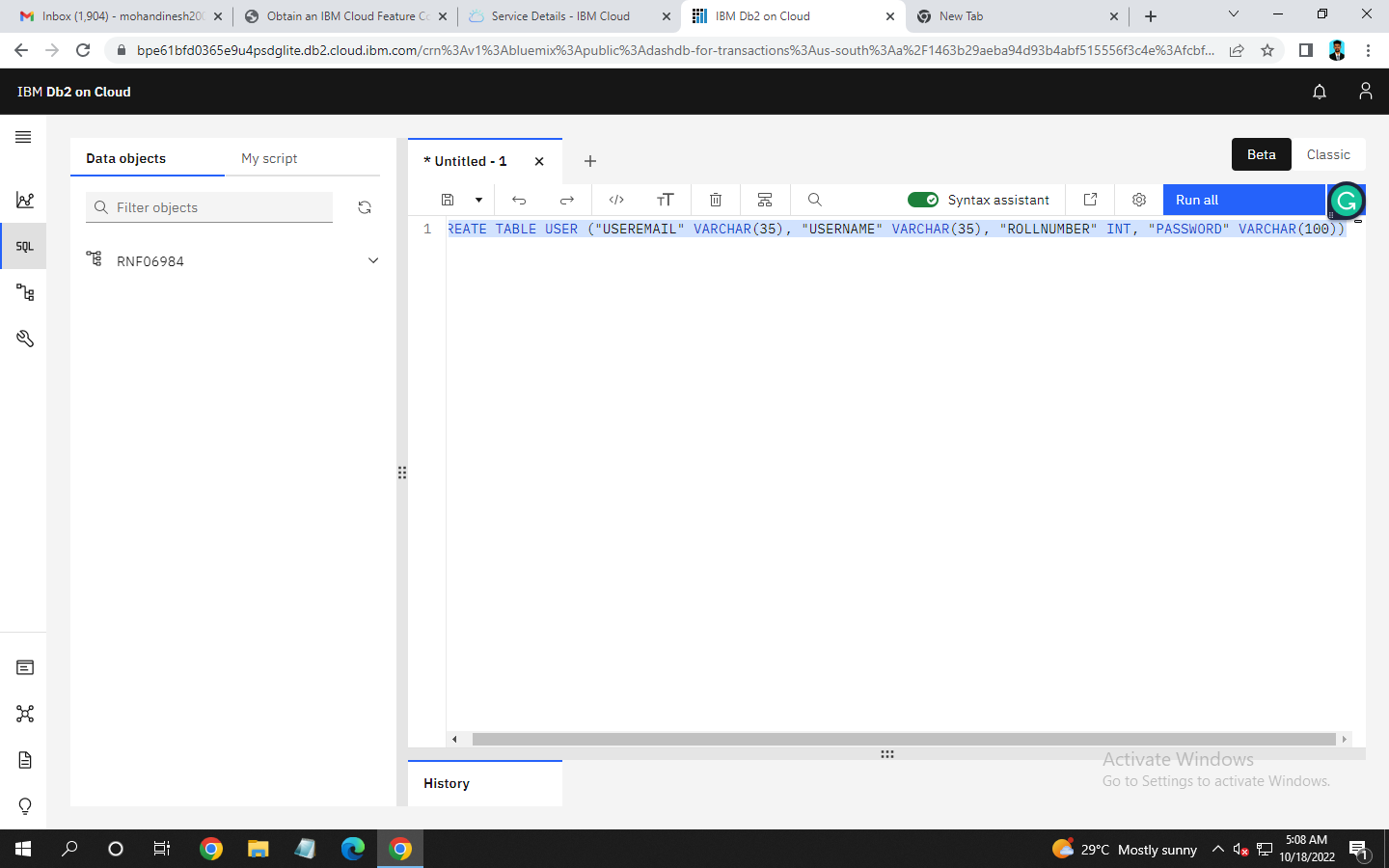
Assignment -2

Connect with Database Assignment

|  |  |
| --- | --- |
| Assignment Date | 09 October 2022 |
| Student Name | Mohan P |
| Student Roll Number | 621319104031 |
| Maximum Marks | 2 Marks |

Question-1:

1. Create User table with user with email , username, roll number, password.



1. Perform UPDATE, DELETE Queries with User table

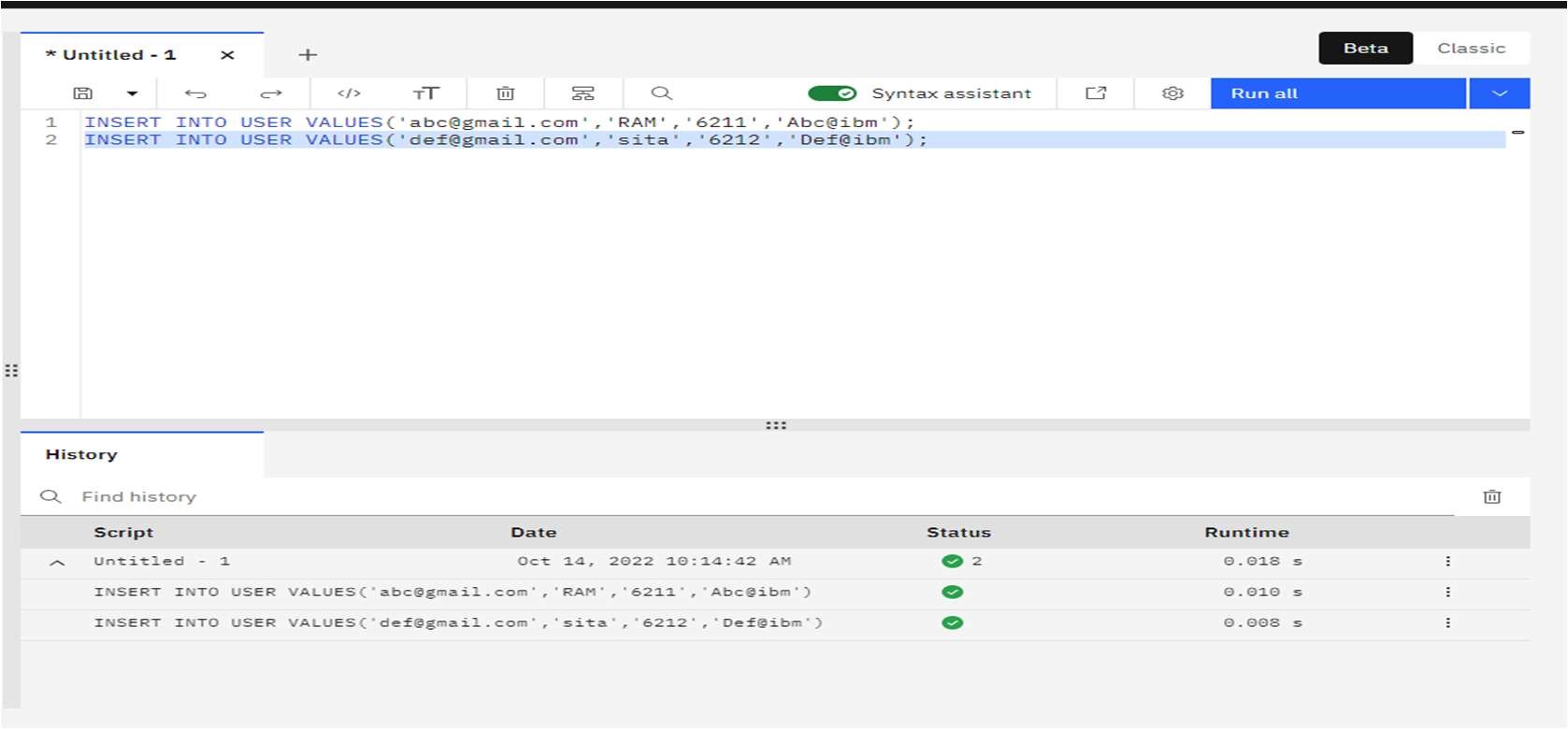
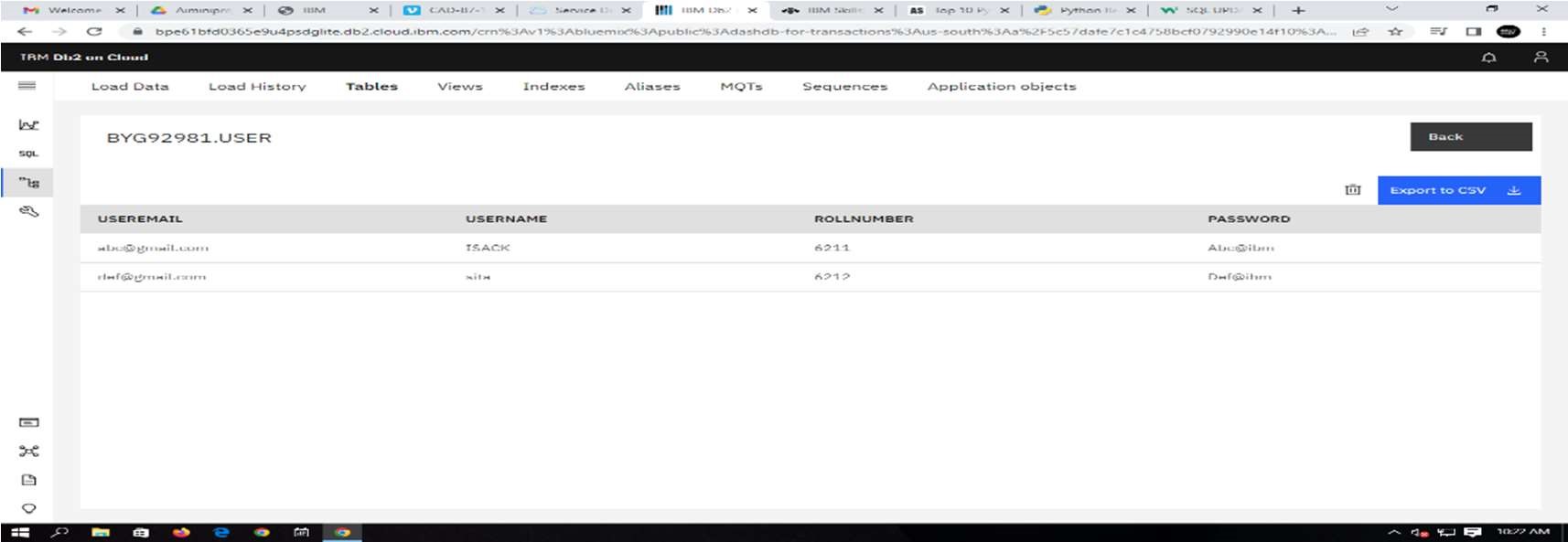


Table View:



UPDATE:

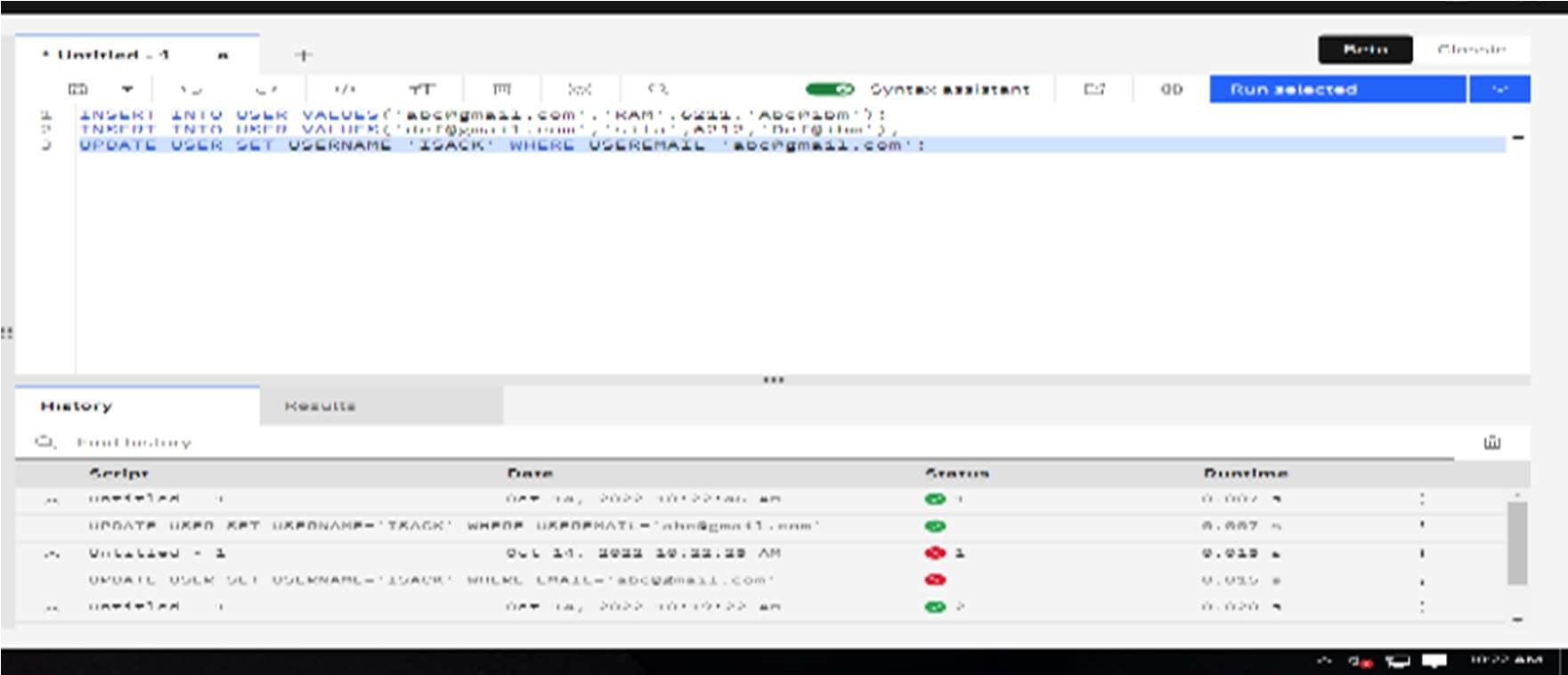
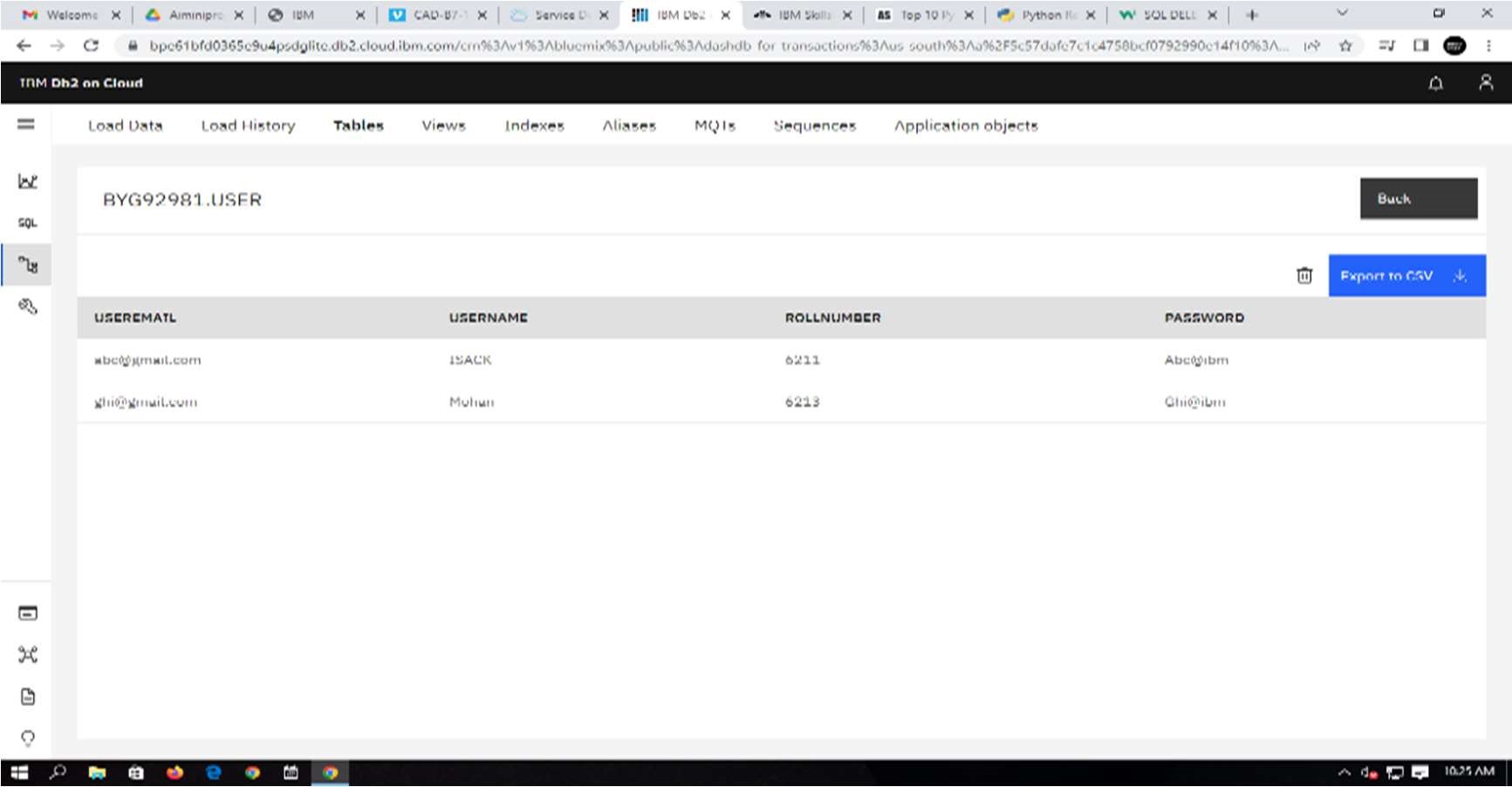


Table View:



DELETE:

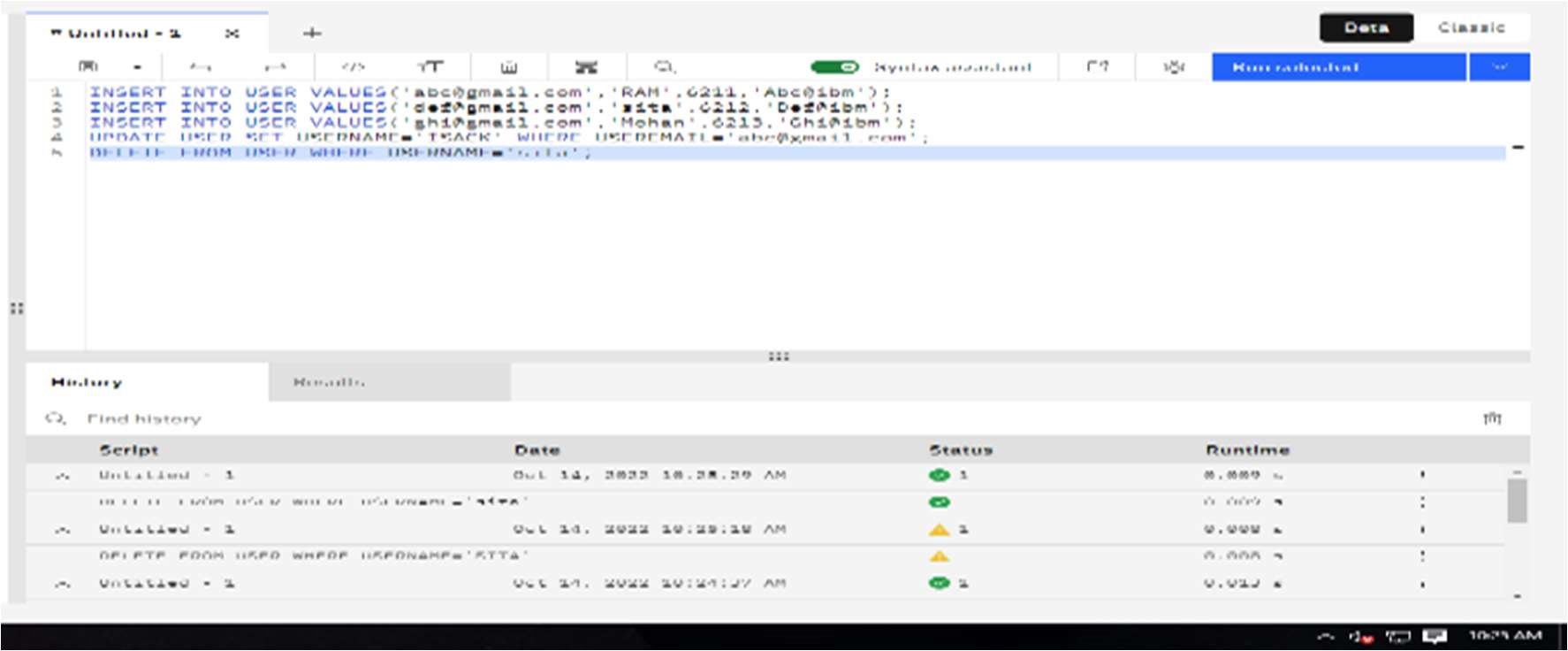
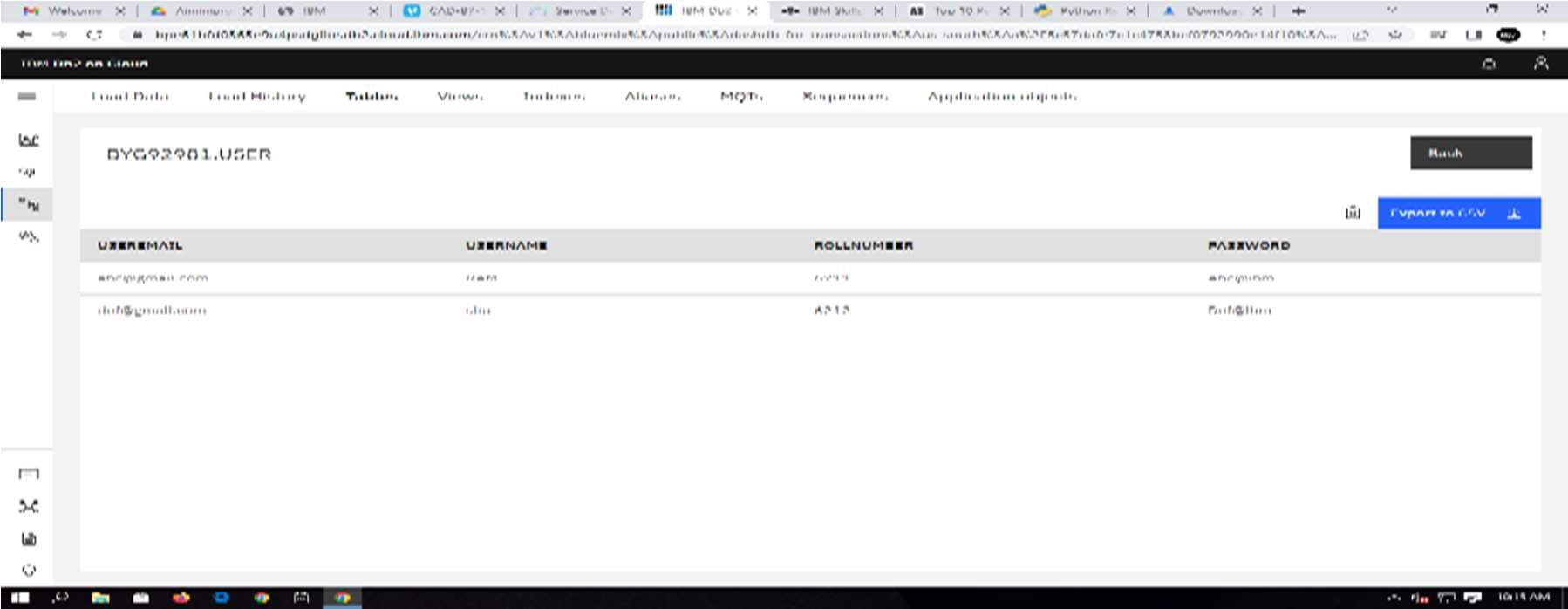


TABLE View:



1. Connect python with db2.

Solution:

import ibm\_db conn = ibm\_db.connect("DATABASE=bludb;HOSTNAME=6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=30376;SECURITY=SSL;SSLS erverCertificate=DigiCertGl obalRootCA.crt;PROTOCOL=TCPIP;UID= rnf06984

;PWD=” VWqiPBgxELVtAn32

",'','')

1. Create a flask app with the registration page. Login page and the welcome page. By default load the registration page once the user enters all the fields, store the data in database and navigate to login page. Authenticate user username and password. If the user is valid so the welcome page.

Solution:

app.py from flask import Flask, render\_template, request, redirect, url\_for, session

import ibm\_db import bcrypt conn =

ibm\_db.connect("DATABASE=bludb;HOSTNAME=;PORT=;SECURITY=SSL;SSLServerCertificate=DigiCer tGlobalRootCA.crt;UID=;PWD=",'','') # url\_for('static', filename='style.css')

app = Flask( name )

app.secret\_key = 'C21FGSBAPOK43K5VSIDFB2'

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

if 'email' not in session:

return redirect(url\_for('login'))

return render\_template('home.html',name='Home')

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

def register(): if request.method == 'POST': email = request.form['email'] username = request.form['username'] rollNo = request.form['rollNo']

password = request.form['password']

if not email or not username or not rollNo or not password:

return render\_template('register.html',error='Please fill all fields')

hash=bcrypt.hashpw(password.encode('utf-8'),bcrypt.gensalt())

query = "SELECT \* FROM USER WHERE email=? OR rollNo=?" stmt = ibm\_db.prepare(conn, query) ibm\_db.bind\_param(stmt,1,email) ibm\_db.bind\_param(stmt,2,rollNo) ibm\_db.execute(stmt) isUser = ibm\_db.fetch\_assoc(stmt)

if not isUser:

insert\_sql = "INSERT INTO User(username,email,PASSWORD,rollNo) VALUES (?,?,?,?)" prep\_stmt = ibm\_db.prepare(conn, insert\_sql) ibm\_db.bind\_param(prep\_stmt, 1, username) ibm\_db.bind\_param(prep\_stmt, 2, email) ibm\_db.bind\_param(prep\_stmt, 3, hash) ibm\_db.bind\_param(prep\_stmt, 4, rollNo) ibm\_db.execute(prep\_stmt)

return render\_template('register.html',success="You can login")

else: return render\_template('register.html',error='Invalid Credentials') return render\_template('register.html',name='Home')

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

if request.method == 'POST': email = request.form['email']

password = request.form['password']

if not email or not password:

return render\_template('login.html',error='Please fill all fields')

query = "SELECT \* FROM USER WHERE email=?" stmt = ibm\_db.prepare(conn, query) ibm\_db.bind\_param(stmt,1,email) ibm\_db.execute(stmt) isUser = ibm\_db.fetch\_assoc(stmt) print(isUser,password)

if not isUser:

return render\_template('login.html',error='Invalid Credentials')

isPasswordMatch = bcrypt.checkpw(password.encode('utf-8'),isUser['PASSWORD'].encode('utf-

8'))

if not isPasswordMatch:

return render\_template('login.html',error='Invalid Credentials')

session['email'] = isUser['EMAIL']

return redirect(url\_for('home'))

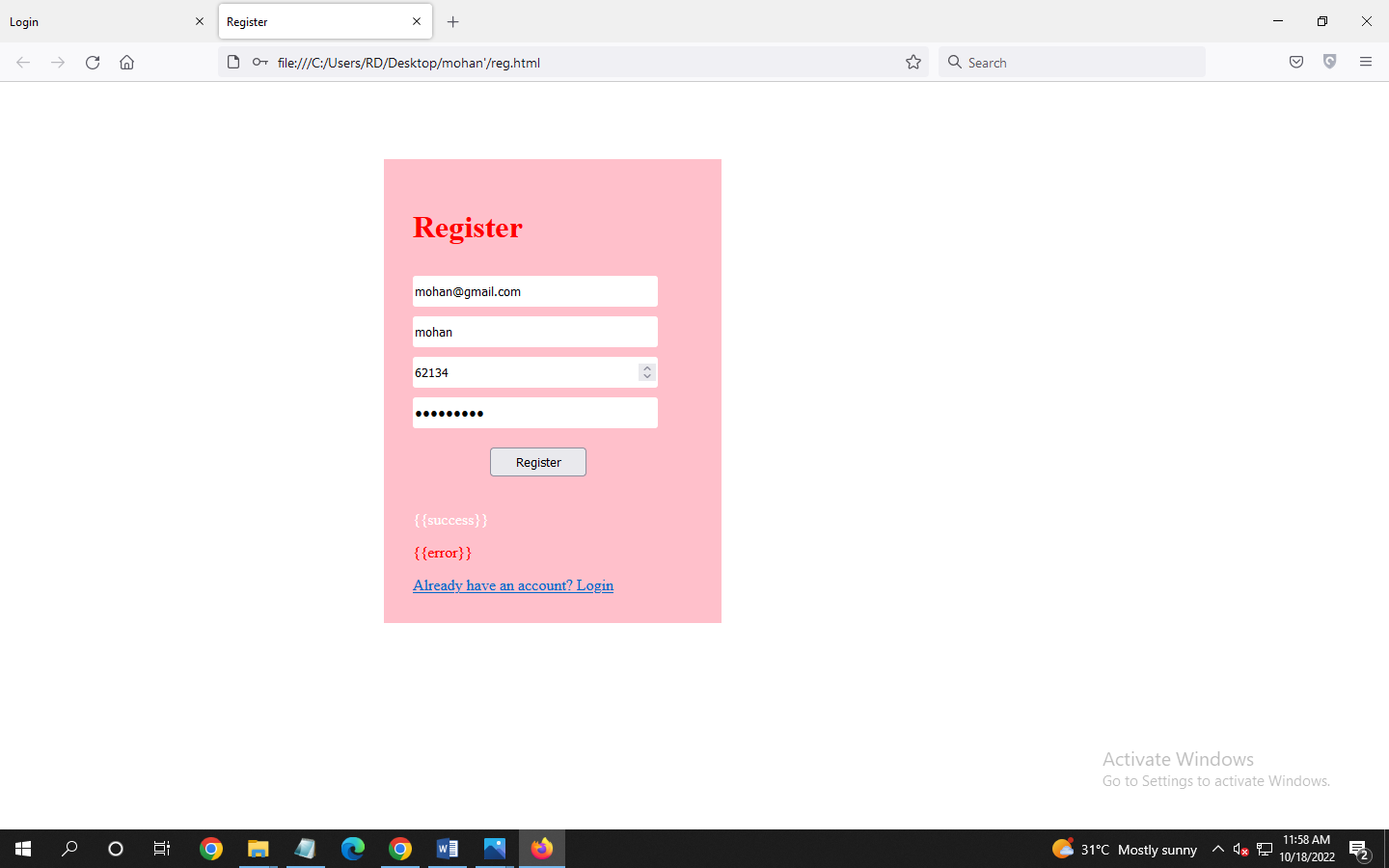
return render\_template('login.html',name='Home')

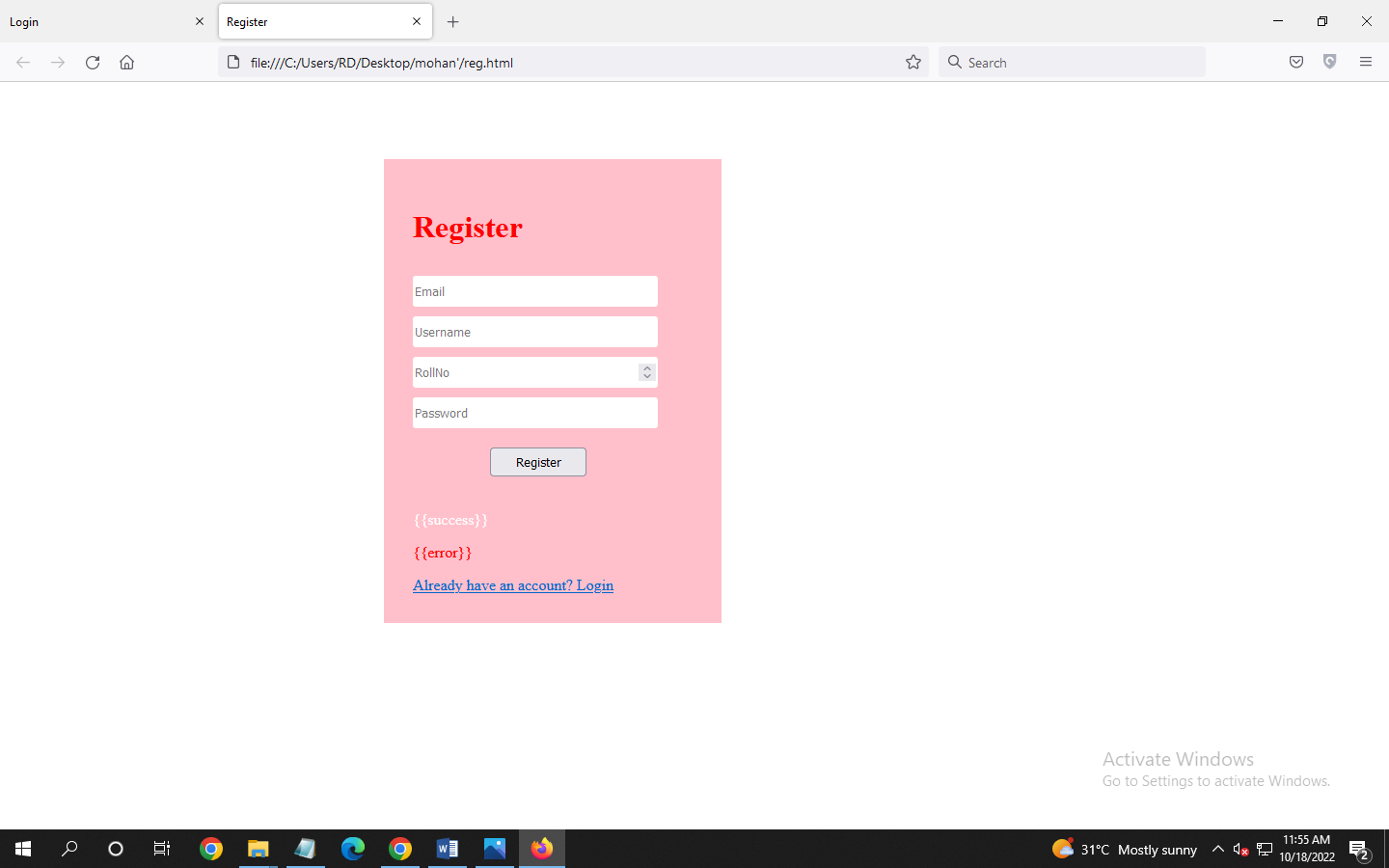
@app.route('/logout') def logout():

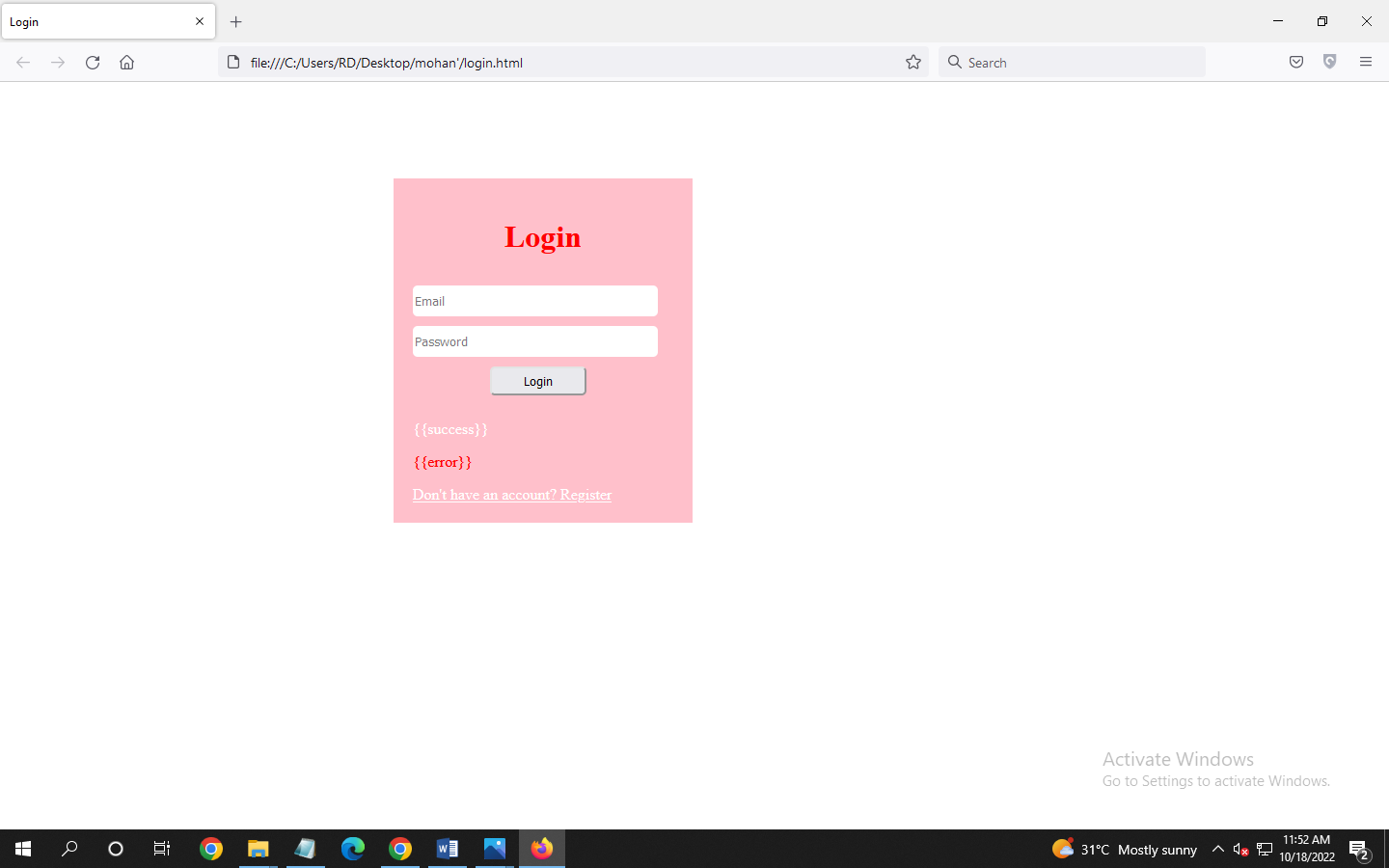
session.pop('email', None)

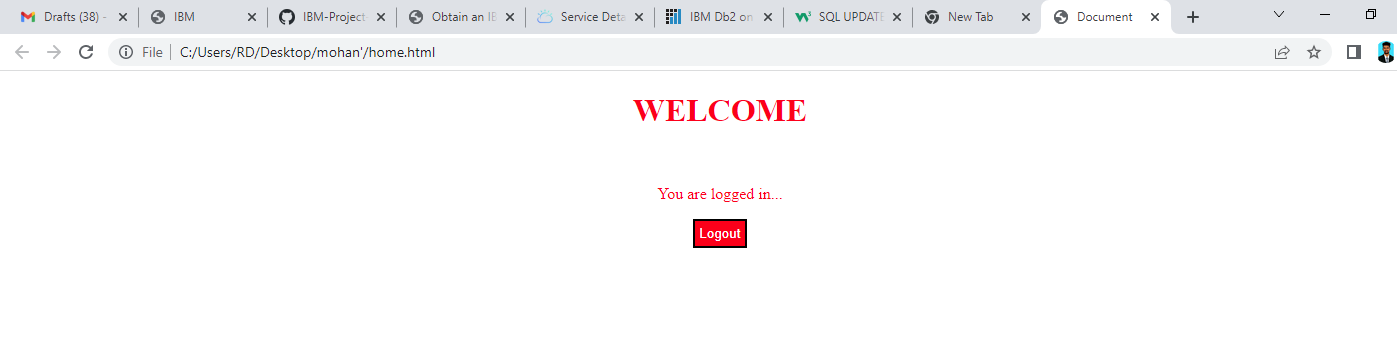
return redirect(url\_for('login'))

OUTPUT:









Database:

