

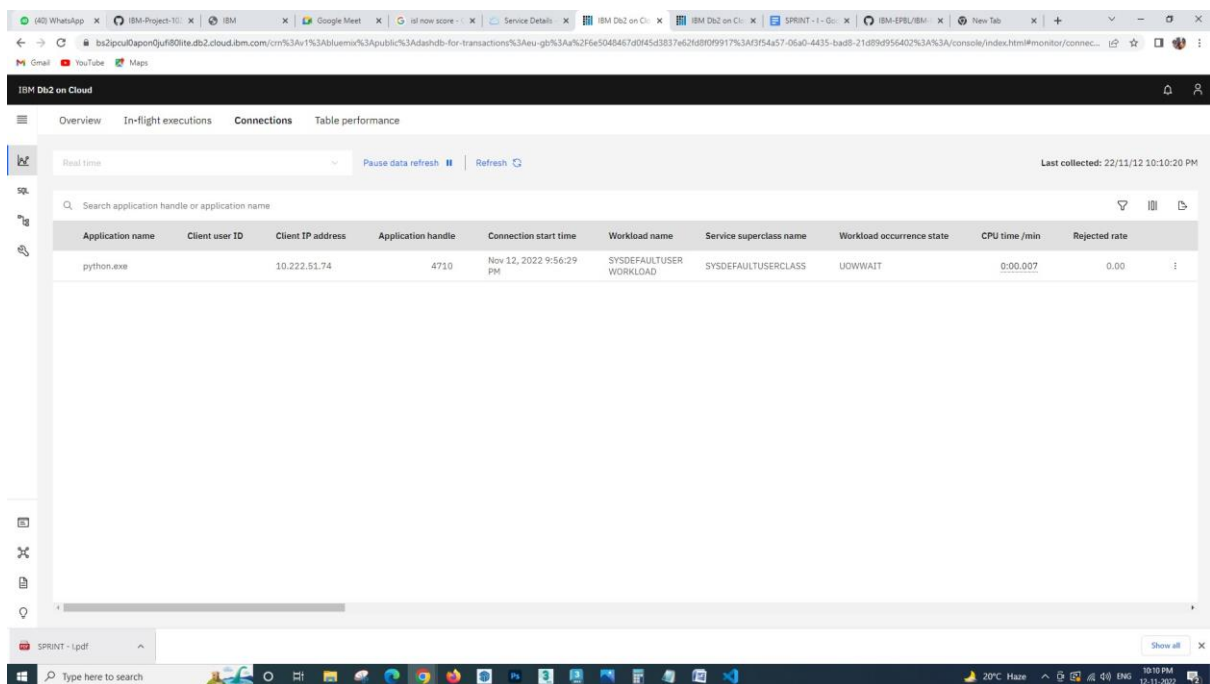
IMPLEMENTING WEB APPLICATION

Create IBM_DB2 And Connect with Python:

Team Id : PNT2022TMID45514

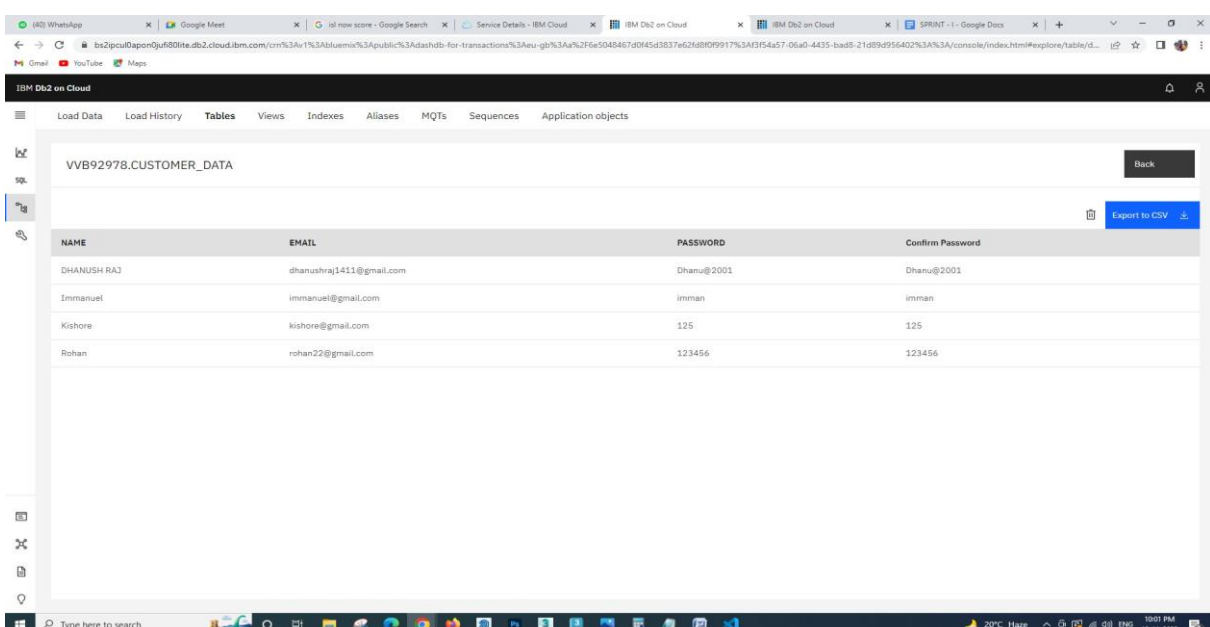
Project Name: Inventory Management System for Retailers

Connected the IBM_db2 using Python:



The screenshot shows the IBM Db2 on Cloud console interface. The 'Connections' tab is selected, displaying a table of active connections. The table has columns for Application name, Client user ID, Client IP address, Application handle, Connection start time, Workload name, Service superclass name, Workload occurrence state, CPU time /min, and Rejected rate. One connection is visible for 'python.exe'.

Application name	Client user ID	Client IP address	Application handle	Connection start time	Workload name	Service superclass name	Workload occurrence state	CPU time /min	Rejected rate
python.exe		10.222.51.74	4710	Nov 12, 2022 9:56:29 PM	SYSDEFAULTUSER WORKLOAD	SYSDEFAULTUSERCLASS	UOWWAIT	0:00.007	0.00



The screenshot shows the IBM Db2 on Cloud console interface with the 'Tables' tab selected. A table named 'VVB92978.CUSTOMER_DATA' is displayed. The table has columns for NAME, EMAIL, PASSWORD, and Confirm Password. The data includes four rows of customer information.

NAME	EMAIL	PASSWORD	Confirm Password
DHANUSH RAJ	dhanushraj1411@gmail.com	Dhanu@2001	Dhanu@2001
Immanuel	immanuel@gmail.com	imman	imman
Kishore	kishore@gmail.com	125	125
Rohan	rohan22@gmail.com	123456	123456

Source Code :

The screenshot displays the Visual Studio Code interface with a Python Flask application. The main editor window shows the code for `app.py`, which includes imports for `Flask`, `render_template`, and `url_for`. The code defines several routes: `home`, `home2`, `login`, `signup`, and `adminlogin`. The file explorer on the left shows the project structure, including folders like `static`, `templates`, and `files`. The terminal at the bottom shows the command `python app.py` being executed, and the output indicates that the application is running on `http://127.0.0.1:5000`.

The screenshot displays the Visual Studio Code editor interface. The Explorer pane on the left shows the project structure, including files like `app.py`, `index.html`, `login.html`, `signup.html`, `index.css`, `msg.html`, `static`, `ac_icon.webp`, `finalwebp`, `indexcss`, `logowebp`, `simplecss`, `stylecss`, `templates`, `index.html`, `login.html`, `msg.html`, `signup.html`, `app.py`, `DebugGlobalRootCA.crt`, and `README.md`. The main editor area shows the `app.py` file, which contains the following code:

```
from flask import Flask, request, redirect, url_for, session, jsonify
from flask_security import Security, SQLAlchemyUserDatastore, login_required, logout_user, login_user, create_user, get_user
from flask_security.extensions import RecaptchaExtension
from flask_wtf import FlaskForm
from wtforms import StringField, PasswordField, SubmitField
from wtforms.validators import Length, EqualTo, Email, DataRequired

app = Flask(__name__)
app.config['SECRET_KEY'] = 'secret-key'
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///data.db'
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
app.config['SECURITY_PASSWORD_SALT'] = 'secret-salt'

db = SQLAlchemy(app)
security = Security(app, db)

class SignupForm(FlaskForm):
    name = StringField('Name')
    email = StringField('Email')
    password = PasswordField('Password')
    confirm_password = PasswordField('Confirm Password')
    submit = SubmitField('Sign Up')

class LoginForm(FlaskForm):
    email = StringField('Email')
    password = PasswordField('Password')
    submit = SubmitField('Log In')

@app.route('/signup', methods=['POST', 'GET'])
def signup():
    if request.method == 'POST':
        form = SignupForm()
        if form.validate():
            name = request.form['name']
            email = request.form['email']
            password = request.form['password']
            confirm_password = request.form['confirm_password']

            user = security.create_user(email=email, name=name, password=password)
            session['user_id'] = user.id
            return redirect(url_for('index'))
        else:
            return render_template('signup.html', form=form)
    else:
        return render_template('signup.html')

@app.route('/login', methods=['POST', 'GET'])
def login():
    if request.method == 'POST':
        form = LoginForm()
        if form.validate():
            email = request.form['email']
            password = request.form['password']
            user = security.get_user(email=email)
            if user and security.verify_password(password, user.password):
                login_user(user)
                return redirect(url_for('index'))
            else:
                return render_template('login.html', form=form)
        else:
            return render_template('login.html')
    else:
        return render_template('login.html')
```

The terminal at the bottom shows the command `python app.py` being executed, and the output indicates that the application is running successfully on `http://127.0.0.1:5000`.