IMPLEMENTING WEB APPLICATION

CREATE IBM DB2 AND CONNECT WITH PYTHON

Date	25th October 2022
Team ID	PNT2022TMID27406
Project Name	News Tracker Application

Views.py

```
from app import app
from flask import Flask, redirect, render_template, request, session, url_for
import re
import ibm_db
from .request import businessArticles, entArticles, get_news_source,
healthArticles, publishedArticles, randomArticles, scienceArticles,
sportArticles, techArticles, topHeadlines

conn=ibm_db.connect("DATABASE=bludb; HOSTNAME=824dfd4d-99de-440d-9991-
629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=30119;SECURIT
Y=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=prg16100;PWD=3Tdg3EFQT
g7f08bD",' ',' ')

#Connecting IBM DB2 with Python

@app.route('/')
def home():
    articles = publishedArticles()
    return  render_template('home.html', articles = articles,x="login")

@app.route('/headlines')
def headlines():
    headlines = topHeadlines()
    return  render_template('headlines.html', headlines = headlines)

@app.route('/articles')
def articles():
    random = randomArticles()
    return  render_template('articles.html', random = random)

@app.route('/sources')
def sources():
    newsSource = get_news_source()
    return  render_template('sources.html', newsSource = newsSource)

@app.route('/category/business')
def business():
    sources = businessArticles()
```

```
    return  render_template('business.html', sources = sources)

@app.route('/category/tech')
def tech():
    sources = techArticles()
    return  render_template('tech.html', sources = sources)

@app.route('/category/entertainment')
def entertainment():
    sources = entArticles()
    return  render_template('entertainment.html', sources = sources)

@app.route('/category/science')
def science():
    sources = scienceArticles()
    return  render_template('science.html', sources = sources)

@app.route('/category/sports')
def sports():
    sources = sportArticles()
    return  render_template('sport.html', sources = sources)

@app.route('/category/health')
def health():
    sources = healthArticles()
    return  render_template('health.html', sources = sources)

@app.route('/login')
def login():
    return render_template('index.html')

@app.route('/logout')
def logout():
    articles = publishedArticles()
    return render_template('home.html', articles =
articles,x="Login/Register")

@app.route('/login',methods = ['POST'])
def getUser():
    if request.method == 'POST':
        user = request.form['uname']
        password = request.form['upwd']
        sql = "SELECT * FROM data where username = ?"
        stmt = ibm_db.prepare(conn, sql)
        email = user
        # Explicitly bind parameters
        ibm_db.bind_param(stmt, 1,user)
```

```
        ibm_db.execute(stmt)
        dictionary = ibm_db.fetch_assoc(stmt)
        pwd = dictionary["PASSWORD"]
        if password != pwd:
            return render_template('error.html')
        articles = publishedArticles()
        return  render_template('home.html', articles = articles,x=None)

@app.route('/signup',methods = ['POST'])
def storedUser():
    if request.method == 'POST':
        username = request.form['username']
        mail = request.form['mail']
        npwd = request.form['npwd']
        cpwd = request.form['cpwd']

        res = username + mail + npwd +cpwd

        if npwd != cpwd:
            return render_template('index.html')
        sql = "INSERT INTO data (UserName,Email,password,confirmpassword)
VALUES(?,?,?,?);"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, username)
        ibm_db.bind_param(stmt, 2, mail)
        ibm_db.bind_param(stmt, 3, npwd)
        ibm_db.bind_param(stmt, 4, cpwd)
        ibm_db.execute(stmt)
    return render_template('index.html')
```