#Import Flask Library

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

import pymysql.cursors

import hashlib

import datetime

#Initialize the app from Flask

app = Flask(\_\_name\_\_)

#Configure MySQL

conn = pymysql.connect(host='localhost',

port = 3306,

user='root',

password='',

db='PriCoSha',

charset='utf8mb4',

cursorclass=pymysql.cursors.DictCursor)

#Define a route to hello function

@app.route('/')

def hello():

return render\_template('index.html')

#Define route for login

@app.route('/login')

def login():

return render\_template('login.html')

#Define route for register

@app.route('/register')

def register():

return render\_template('register.html')

#Authenticates the login

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

def loginAuth():

#grabs information from the forms

email = request.form['email']

password = request.form['password']

hash\_pass = hashlib.md5(password.encode('utf\_8')).hexdigest()

#cursor used to send queries

cursor = conn.cursor()

#executes query

query = 'SELECT \* FROM Person WHERE email = %s and password = %s'

cursor.execute(query, (email, hash\_pass))

#stores the results in a variable

data = cursor.fetchone()

#use fetchall() if you are expecting more than 1 data row

cursor.close()

error = None

if(data):

#creates a session for the the user

#session is a built in

session['email'] = email

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

else:

#returns an error message to the html page

error = 'Invalid login or email'

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

#Authenticates the register

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

def registerAuth():

#grabs information from the forms

email = request.form['email']

password = request.form['password']

hash\_pass = hashlib.md5(password.encode('utf\_8')).hexdigest()

#cursor used to send queries

cursor = conn.cursor()

#executes query

query = 'SELECT \* FROM Person WHERE email = %s'

cursor.execute(query, (email))

#stores the results in a variable

data = cursor.fetchone()

#use fetchall() if you are expecting more than 1 data row

error = None

if(data):

#If the previous query returns data, then user exists

error = "This user already exists"

return render\_template('register.html', error = error)

else:

ins = 'INSERT INTO person(email, password) VALUES(%s, %s)'

cursor.execute(ins, (email, hash\_pass))

conn.commit()

cursor.close()

return render\_template('index.html')

@app.route('/home')

def home():

user = session['email']

cursor = conn.cursor();

query = 'SELECT post\_time, item\_id FROM contentitem WHERE email = %s ORDER BY post\_time DESC'

cursor.execute(query, (user))

data = cursor.fetchall()

cursor.close()

return render\_template('home.html', email=user, posts=data)

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

def post():

email = session['email']

cursor = conn.cursor();

blog = request.form['blog']

postime = datetime.datetime.now();

is\_public = request.form.get('is\_pub')

is\_pub = 0

if(is\_public):

is\_pub = 1

query = 'INSERT INTO contentitem (item\_name, email, post\_time, is\_pub) VALUES(%s, %s, %s, %s)'

cursor.execute(query, (blog, email, postime, int(is\_pub)))

conn.commit()

cursor.close()

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

@app.route('/select\_blogger')

def select\_blogger():

#check that user is logged in

#username = session['username']

#should throw exception if username not found

cursor = conn.cursor();

query = 'SELECT DISTINCT email FROM contentitem'

cursor.execute(query)

data = cursor.fetchall()

cursor.close()

return render\_template('select\_blogger.html', user\_list=data)

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

def show\_posts():

poster = request.args['poster']

cursor = conn.cursor();

query = 'SELECT post\_time, item\_id FROM contentitem WHERE email = %s ORDER BY post\_time DESC'

cursor.execute(query, poster)

data = cursor.fetchall()

cursor.close()

return render\_template('show\_posts.html', poster\_name=poster, posts=data)

@app.route('/logout')

def logout():

session.pop('email')

return redirect('/')

app.secret\_key = 'some key that you will never guess'

#Run the app on localhost port 5000

#debug = True -> you don't have to restart flask

#for changes to go through, TURN OFF FOR PRODUCTION

if \_\_name\_\_ == "\_\_main\_\_":

app.run('127.0.0.1', 5000, debug = True)