PROJECT 2: AWS

Start Assignment

- Due Sunday by 11:59pm
- Points 10
- Submitting a website url or a file upload
- Attempts 0
- Allowed Attempts 5
- Available Sep 24 at 2pm - Oct 8 at 11:59pm

Total Points : 10 (Extra Credit: 2)

Requirements:

1. Launch an **EC2 instance** (any platform accepted but should be accessible through the internet). (**Points: 2**)

(Screenshots of each major step (AWS portal setups, EC2 creation, Amazon Machine Image (AMI), Key pair (login), Network settings etc.)

2. Configure **web server** and **SQLite3 Database table** (any variation, but help is available for limited variations). (**Points: 3**)

(Screenshots of Flask application setup in the EC2 instance, Install mod_wsgi for Apache, Install Python, Pip, and Flask, SQLite3 Database table etc.)

3. Design an **interactive web page** to do the following: (**Points: 5, broken down as follows**)

a. Create a registration page with (store it): (**Points: 1/5**)

i. Username

ii. Password

b. Accept users basic details (store it): (**Points: 1/5**)

i. First name

ii. Last name

iii. Email

c. Upon submission redirect to next page (**Points: 1/5**)

i. Display information accepted in step 3b

d. Should ask for username and password to retrieve user information. (**Points: 2/5**)

Submit the AWS URL for the Webpage (Example http://ec2-x-x-x-xxxx..us-east-2.compute.amazonaws.com)

Share the code in GitHub or as an attachment. **(2 points will be deducted for not sharing the code) (Example https://github.com/UserName)**

Extra Credit (Points : 2)

1. Upload the file **Limerick-1.txt (https://uc.instructure.com/courses/1712022/files/184312741?wrap=1)** **(https://uc.instructure.com/courses/1712022/files/184312741/download?download_frd=1)** . with above designed form.
2. **(https://uc.instructure.com/courses/1587528/files/164327703?wrap=1)** Store it.
3. Display the count of words in the file on page designed in step#3c and provide link to download it.
4. Display all the information from extra credit step#3 upon relogin.

Hints: Use information in the class slides.

- Use the instruction on **Running a Flask app on AWS EC2 (https://www.datasciencebytes.com/bytes/2015/02/24/running-a-flask-app-on-aws-ec2/)** for the assignment.

Hint 1: The ~~Ubuntu Server *14.04 LTS* (HVM) AMI~~, but is not available as a Free Tier in AWS anymore. Please Choose the **"Free tier eligible"** Amazon Machine Image (AMI): Example Ubuntu Server 24.04 LTS.

Hint 2: **Ubuntu Server 24.04 LTS only supports python3.**

~~$ sudo apt-get install python-pip will fail, due to Package python-pip being deprecated~~

Here are the overrides to https://www.datasciencebytes.com/bytes/2015/02/24/running-a-flask-app-on-aws-ec2:

1.1 Launch an EC2 instance: Ubuntu Server 24.04 LTS (HVM)

2.1 Install the apache webserver and mod_wsgi.

```
sudo apt-get update

sudo apt-get install apache2

sudo apt install libapache2-mod-wsgi-py3
```

2.2 Install Flask using the pip tool

```
sudo apt install python3-pip

sudo apt install python3-flask

chmod 755 /home/ubuntu/
```

5. Test configuration.

```
tail /var/log/apache2/error.log
```

Hint 3: Instruction in Using Flask to answer SQL queries (https://www.datasciencebytes.com/bytes/2015/02/28/using-flask-to-answer-sql-queries/) is incomplete.

i. Install SQLite3: SQLite3 is typically pre-installed. If not, use:

```
sudo apt install sqlite3 -y
```

ii. Verify SQLite3 Installation:

```
sqlite3 --version
```

iii. Create a New SQLite3 Database (or open an existing one)

```
sqlite3 mydatabase.db
```

iv. Create a Table:

```
CREATE TABLE users (
   id INTEGER PRIMARY KEY AUTOINCREMENT,
   username TEXT NOT NULL,
   password TEXT NOT NULL,
   email TEXT NOT NULL,

   ...
);
```

v. Insert Data into the Table:

```
INSERT INTO users (username, password, email) VALUES ('testuser', 'password123', 'testuser@example.com');
```

vi. Retrieve Data from the Table:

```
SELECT * FROM users;
```

vii. Exit SQLite3:

```
.exit
```

viii. Example only of a Flask application file:

```
vi flaskapp.py
```

Add the following content (Example only):

```
from flask import Flask, render_template, request, redirect, url_for
import sqlite3

app = Flask(__name__)

# SQLite setup
conn = sqlite3.connect('users.db')
c = conn.cursor()
c.execute('''CREATE TABLE IF NOT EXISTS users
          (username TEXT, password TEXT, firstname TEXT, lastname TEXT, email TEXT)''')
conn.commit()
conn.close()

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

@app.route('/register', methods=['POST'])
def register():
   username = request.form['username']
```

```
    password = request.form['password']
    firstname = request.form['firstname']
    lastname = request.form['lastname']
    email = request.form['email']

    conn = sqlite3.connect('users.db')
    c = conn.cursor()
    c.execute("INSERT INTO users (username, password, firstname, lastname, email) VALUES (?, ?, ?, ?, ?)",
              (username, password, firstname, lastname, email))
    conn.commit()
    conn.close()

    return redirect(url_for('profile', username=username))

@app.route('/profile/<username>')
def profile(username):
    conn = sqlite3.connect('users.db')
    c = conn.cursor()
    c.execute("SELECT * FROM users WHERE username=?", (username,))
    user = c.fetchone()
    conn.close()

    return render_template('profile.html', user=user)

if __name__ == '__main__':
    app.run(debug=True)
```

ix. Example of the HTML Files:

In the `templates/` folder, create `register.html`:

```
<form method="POST" action="/register">
    Username: <input type="text" name="username"><br>
    Password: <input type="password" name="password"><br>
    First Name: <input type="text" name="firstname"><br>
    Last Name: <input type="text" name="lastname"><br>
    Email: <input type="text" name="email"><br>
    <input type="submit" value="Register">
</form>
```

x. Submit the AWS URL for the Webpage

Example **http://ec2-x-x-x-xxxx..us-east-2.compute.amazonaws.com (http://ec2-x-x-x-xxxx..us-east-2.compute.amazonaws.com)** . Make sure the network security setup is correct.

xi. Here helpful URLs on deploying WebApp in the Amazon EC2:

Tutorial: Get started with Amazon EC2 Linux instances (https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html)

(https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html) Creating AWS account (https://aws.amazon.com/)

Running a Flask app on AWS EC2 (https://www.datasciencebytes.com/bytes/2015/02/24/running-a-flask-app-on-aws-ec2/)

Using Flask to answer SQL queries (https://www.datasciencebytes.com/bytes/2015/02/28/using-flask-to-answer-sql-queries/)