CLOUD PROJECT 1
DHRUV YADAV
N01650900
0NC

Project Overview
The Sentiment Analysis Flask Server project aims to provide a Flask-based server for sentiment analysis using the AWS Comprehend service. This document provides an overview of the project development and describes each file in detail.

Flask Server Structure
The project consists of the following files and directories:

1. main.py: This file contains the main Flask application code. It defines the routes for handling user requests, such as home page rendering and sentiment analysis.

2. service.py: This file contains the service function get_sentiment() responsible for interacting with the AWS Comprehend service using boto3 to perform sentiment analysis on the provided text.

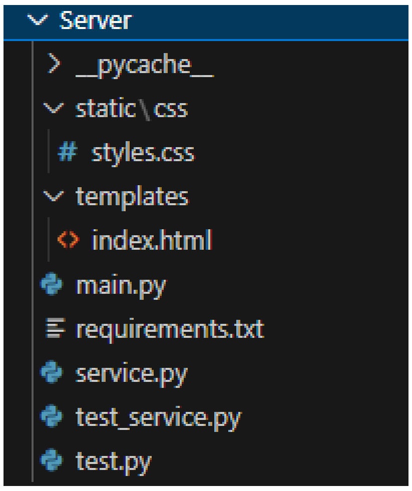
3. test_service.py: his file contains the service function get_sentiment() for testing the website response when run offline. It is identical to service.py but is used specifically for testing purposes when the connection to the AWS Comprehend service (using boto3) cannot be established.

4. test.py: It is a script designed for offline testing of the Sentiment Analysis Flask Server. It sends a POST request to http://127.0.0.1:80/analyze with sample text data and parses the HTML response using BeautifulSoup. It extracts the sentiment and sentiment scores, printing them to the console. This script aids in verifying the server's functionality and response handling.

5. index.html: This HTML template file defines the structure and layout of the web interface for the sentiment analysis server. It includes a form for inputting text and displaying sentiment analysis results. The file can be found within the /templates/ directory.

6. styles.css: This CSS file contains styles for customizing the appearance of the web interface. The file can be found within the /static/css/ directory.

7. requirements.txt: This file lists all the Python dependencies required for running the Flask server. It can be used with pip to install the necessary packages.

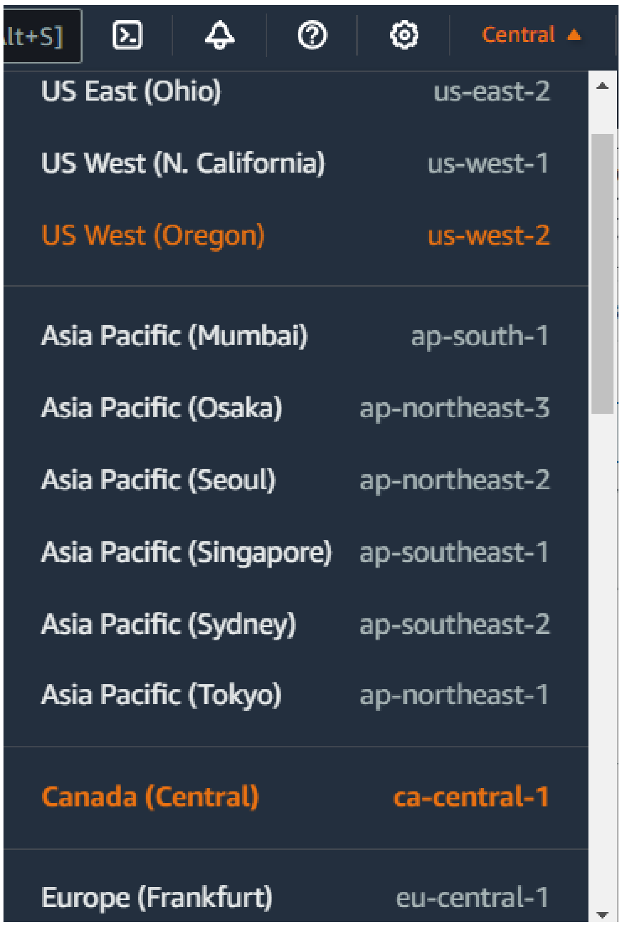


```
Server
  > __pycache__
  static\css
    # styles.css
  templates
    <> index.html
  main.py
  requirements.txt
  service.py
  test_service.py
  test.py
```

Launching a new EC2 Instance

To launch a new EC2 instance with the specified configuration, follow these steps:

1. Sign in to https://myapps.microsoft.com/ using Humber credentials.
2. Open AWS-Fast AI Academic-28.
3. Select Canada (Central) (ca-central-1) as the region.

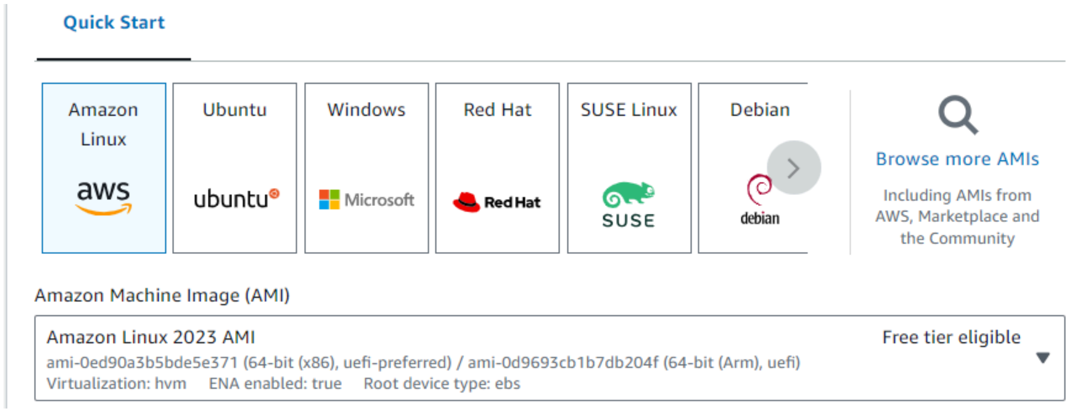


US East (Ohio)	us-east-2
US West (N. California)	us-west-1
US West (Oregon)	us-west-2
Asia Pacific (Mumbai)	ap-south-1
Asia Pacific (Osaka)	ap-northeast-3
Asia Pacific (Seoul)	ap-northeast-2
Asia Pacific (Singapore)	ap-southeast-1
Asia Pacific (Sydney)	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
Canada (Central)	ca-central-1
Europe (Frankfurt)	eu-central-1

4. Navigate to the EC2 Dashboard.

5. Click on the "Launch Instance" button to start the instance creation process.

6. In the "Choose an Amazon Machine Image (AMI)" step, select "Amazon Linux (Amazon Linux 2023 AMI)" as the operating system for your instance.



Amazon Machine Image (AMI)

Amazon Linux 2023 AMI
Free tier eligible
ami-0ed90a3b5bde5e371 (64-bit (x86), uefi-preferred) / ami-0d9693cb1b7db204f (64-bit (Arm), uefi)
Virtualization: hvm ENA enabled: true Root device type: ebs

7. Under “Key pair (login)” section, select create a new key-pair, give an appropriate name and download the .ppk key pair file.

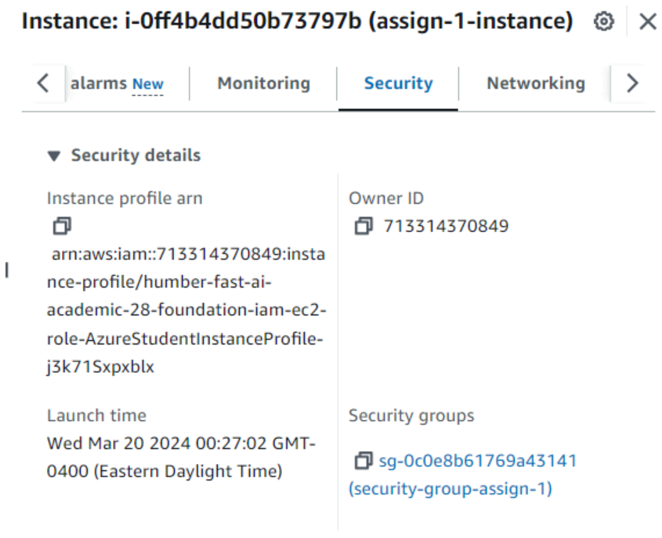
8. Under “Network Settings” section, select subnet sbn-fast-ai-academic-28-public-ca-central-1a.

9. Select Auto-assign public IP to be enabled.(EIP binding is done later)

10. Under “Firewall (security groups)” select Create security group and write a security group name. (Inbound rule for security group is edited later)

11. Click “Launch Instance”.

12. Go to the EC2 dashboard and select the newly created instance. Under “Security” tab select the security group name link to edit the security group’s rules.



Instance: i-0ff4b4dd50b73797b (assign-1-instance)

Security details

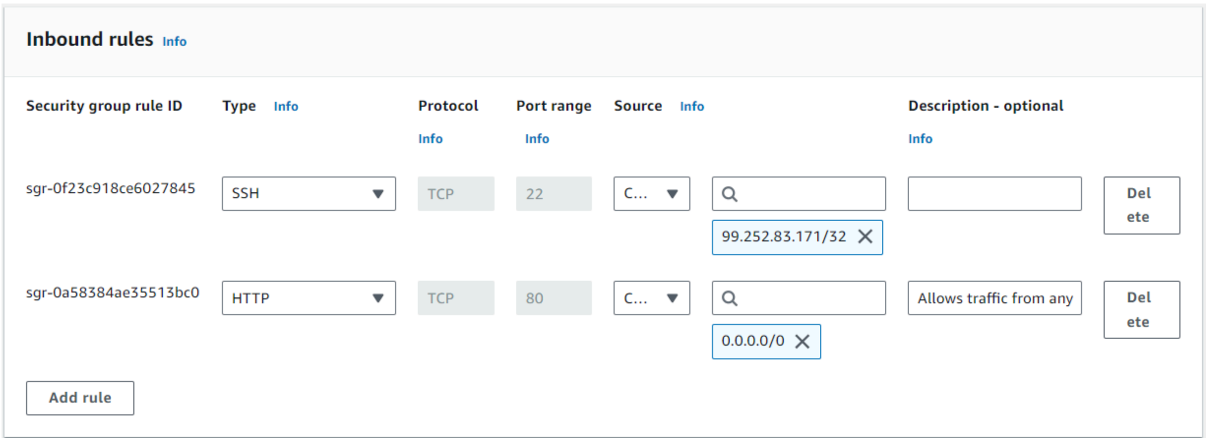
Instance profile arn
arn:aws:iam::713314370849:instance-profile/humber-fast-ai-academic-28-foundation-iam-ec2-role-AzureStudentInstanceProfile-j3k71Sxpxblx

Owner ID
713314370849

Launch time
Wed Mar 20 2024 00:27:02 GMT-0400 (Eastern Daylight Time)

Security groups
sg-0c0e8b61769a43141 (security-group-assign-1)

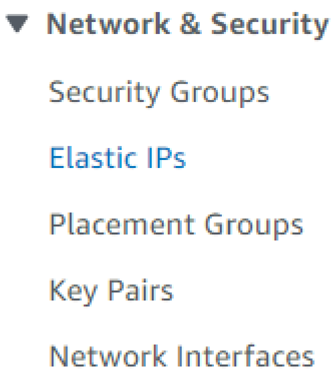
13. Select “Edit Inbound Rules” and add a rule of type - http, port range - 80, source - custom 0.0.0.0/0 and select “Save rules”.



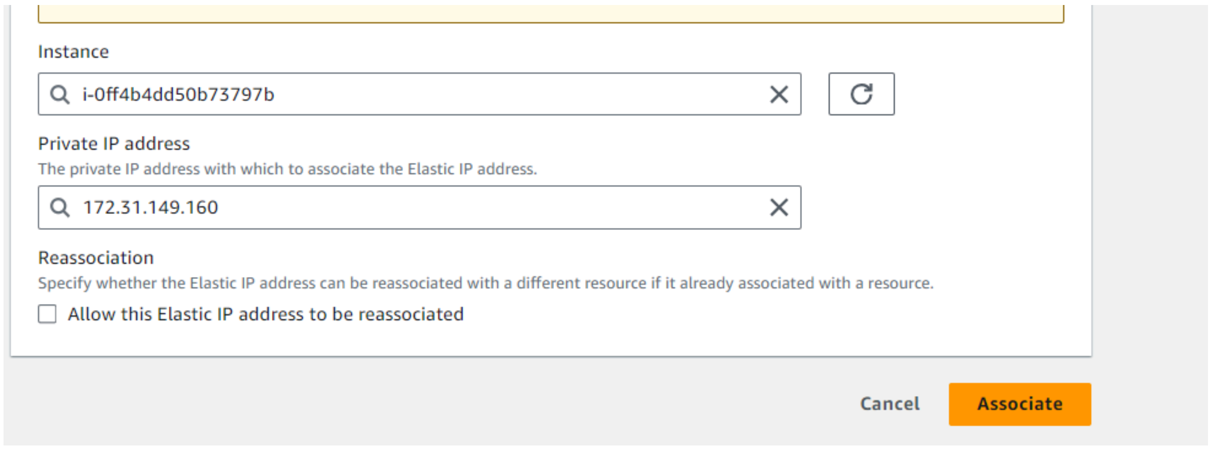
Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-0f23c918ce6027845	SSH	TCP	22	C...	99.252.83.171/32		Delete
sgr-0a58384ae35513bc0	HTTP	TCP	80	C...	0.0.0.0/0	Allows traffic from any	Delete

14. Under section “Network and Security” in EC2 side panel, select Elastic IP.



Select the elastic IP and click Allocate Elastic IP Address. Select the instance and click associate.



Instance

i-0ff4b4dd50b73797b

Private IP address

The private IP address with which to associate the Elastic IP address.

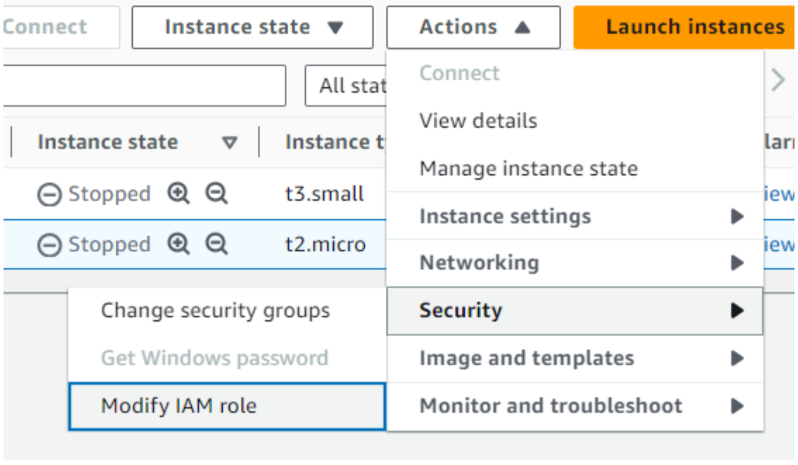
172.31.149.160

Reassociation

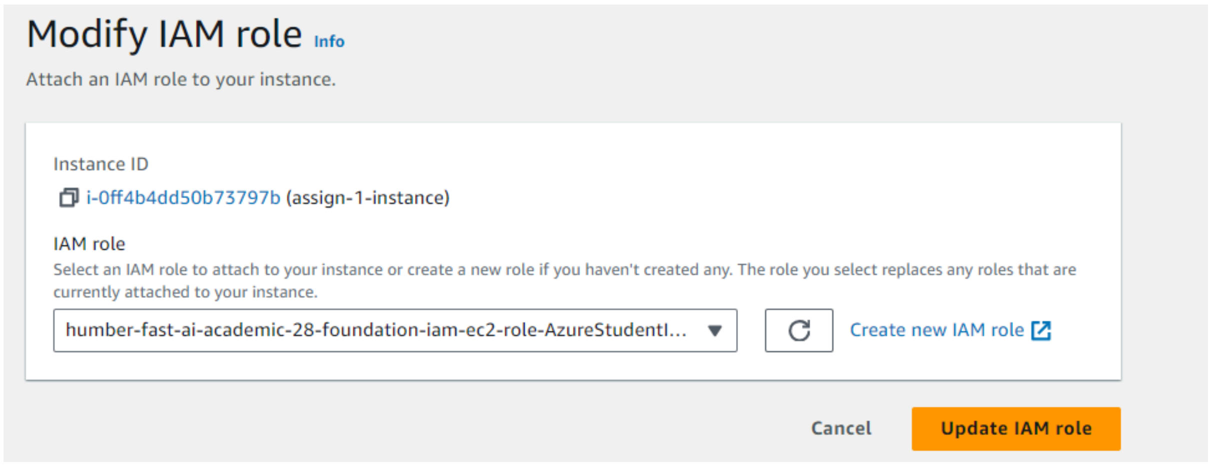
Specify whether the Elastic IP address can be reassociated with a different resource if it already associated with a resource.

☐ Allow this Elastic IP address to be reassociated

15. Again go the the EC2 instances, select the instance. Select Actions > Security > Modify IAM Role.



Select fast-ai-academic-28-Student-EC2 role and click Update IAM Role.


Modify IAM role Info

Attach an IAM role to your instance.

Instance ID

i-0ff4b4dd50b73797b (assign-1-instance)

IAM role

Select an IAM role to attach to your instance or create a new role if you haven't created any. The role you select replaces any roles that are currently attached to your instance.

humber-fast-ai-academic-28-foundation-iam-ec2-role-AzureStudentI...

16. Finally select the EC2 instance and click Start Instance.

Connecting to EC2 Instance

To connect to your EC2 instance using PuTTY, follow these steps:

1. Open PuTTY.

2. Enter your EC2 instance's public IP address in the "Host Name (or IP address)" field.

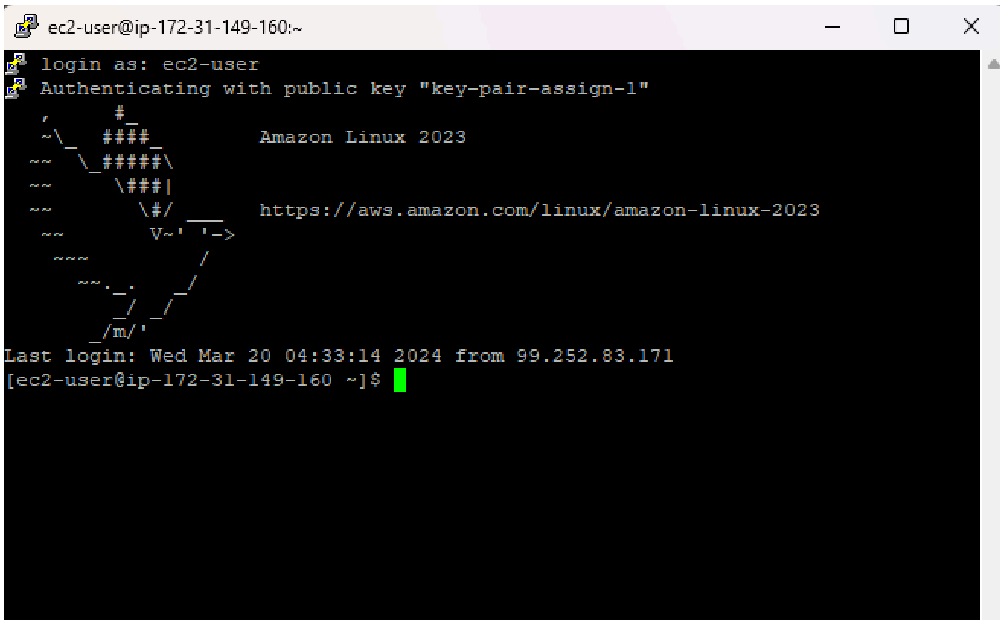
3. Under "Connection," expand "SSH" and click on "Auth."

4. Click on the "Browse" button and select the .ppk private key file.

5. Save these settings by entering a name in the "Saved Sessions" field and clicking the "Save" button.

6. Click on the "Open" button to start the SSH session.

7. Enter username "ec2-user".



```
login as: ec2-user
Authenticating with public key "key-pair-assign-1"
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Wed Mar 20 04:33:14 2024 from 99.252.83.171
[ec2-user@ip-172-31-149-160 ~]$
```

8. Make a new directory called 'project1' using mkdir.

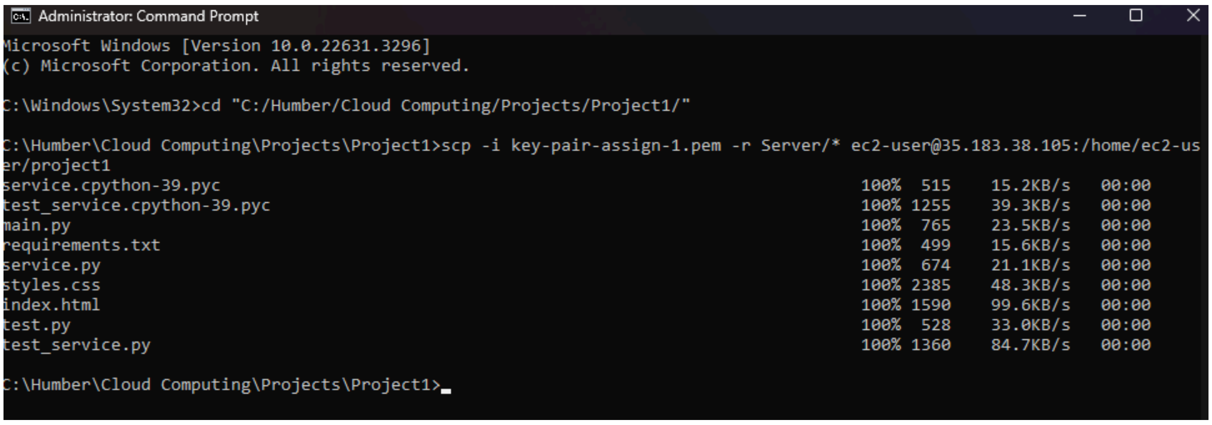
Uploading Server Files

To upload server files to your EC2 instance, you can use SCP (Secure Copy Protocol) method. Below are the steps to upload server files using SCP:

1. Open a terminal or command prompt window on your local machine.

2. Use the cd command to navigate to the project directory on your local machine. (This directory has another subdirectory called /Server/ which contains the actual code).

3. Use the following SCP command to upload files to your EC2 instance:
   scp -i /path/to/private_key.pem -r /path/to/local/folder/*
   ec2-user@<public_ip>:/home/ec2-user/project1



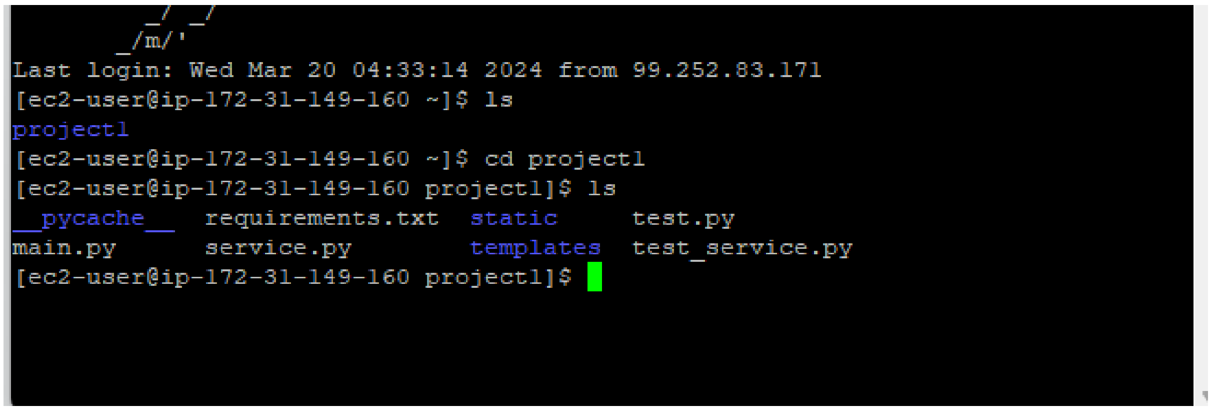
```
Microsoft Windows [Version 10.0.22631.3296]
(c) Microsoft Corporation. All rights reserved.

:\Windows\System32>cd "C:/Humber/Cloud Computing/Projects/Project1/"

:\Humber\Cloud Computing\Projects\Project1>scp -i key-pair-assign-1.pem -r Server/* ec2-user@35.183.38.105:/home/ec2-us
er/project1
service.cpython-39.pyc                                                      100%  515    15.2KB/s   00:00
test_service.cpython-39.pyc                                                 100% 1255    39.3KB/s   00:00
main.py                                                                     100%  765    23.5KB/s   00:00
requirements.txt                                                            100%  499    15.6KB/s   00:00
service.py                                                                  100%  674    21.1KB/s   00:00
styles.css                                                                  100% 2385    48.3KB/s   00:00
index.html                                                                  100% 1590    99.6KB/s   00:00
test.py                                                                     100%  528    33.0KB/s   00:00
test_service.py                                                             100% 1360    84.7KB/s   00:00

:\Humber\Cloud Computing\Projects\Project1>
```

4. Check that files are uploaded in EC2 instance.



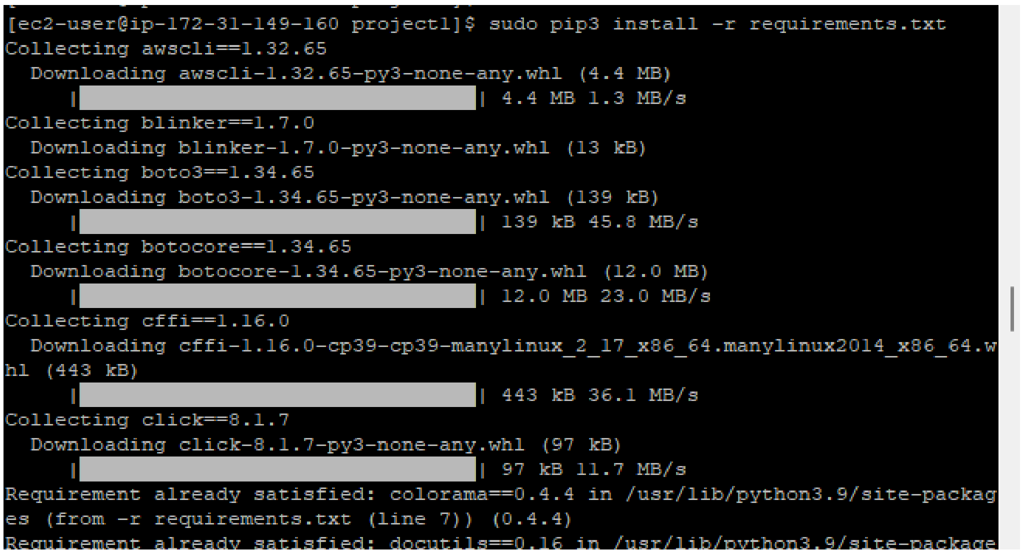
```
     _/ _/
    _/m/'
Last login: Wed Mar 20 04:33:14 2024 from 99.252.83.171
[ec2-user@ip-172-31-149-160 ~]$ ls
project1
[ec2-user@ip-172-31-149-160 ~]$ cd project1
[ec2-user@ip-172-31-149-160 project1]$ ls
__pycache__   requirements.txt  static     test.py
main.py       service.py        templates  test_service.py
[ec2-user@ip-172-31-149-160 project1]$
```

Installing Dependencies

To install dependencies on your EC2 instance, follow these steps:

1. Install Python 3.9 (if not already installed) using the package manager. Use the following commands for Amazon Linux:
   sudo yum update
   sudo yum install python3
   sudo yum install python3-pip

2. Navigate to the /project1/ directory and install the required dependencies using pip:
   sudo pip3 install -r requirements.txt



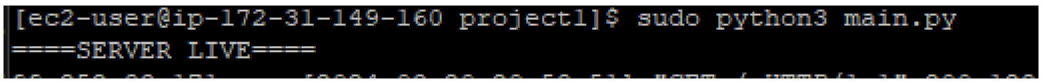
```
[ec2-user@ip-172-31-149-160 project1]$ sudo pip3 install -r requirements.txt
Collecting awscli==1.32.65
  Downloading awscli-1.32.65-py3-none-any.whl (4.4 MB)
     |████████████████████████████████| 4.4 MB 1.3 MB/s
Collecting blinker==1.7.0
  Downloading blinker-1.7.0-py3-none-any.whl (13 kB)
Collecting boto3==1.34.65
  Downloading boto3-1.34.65-py3-none-any.whl (139 kB)
     |████████████████████████████████| 139 kB 45.8 MB/s
Collecting botocore==1.34.65
  Downloading botocore-1.34.65-py3-none-any.whl (12.0 MB)
     |████████████████████████████████| 12.0 MB 23.0 MB/s
Collecting cffi==1.16.0
  Downloading cffi-1.16.0-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.w
hl (443 kB)
     |████████████████████████████████| 443 kB 36.1 MB/s
Collecting click==8.1.7
  Downloading click-8.1.7-py3-none-any.whl (97 kB)
     |████████████████████████████████| 97 kB 11.7 MB/s
Requirement already satisfied: colorama==0.4.4 in /usr/lib/python3.9/site-packag
es (from -r requirements.txt (line 7)) (0.4.4)
Requirement already satisfied: docutils==0.16 in /usr/lib/python3.9/site-package
```

Launching the Server

To launch the server on your EC2 instance, follow these steps:

1. Run the Python script ‘main.py’ that contains your server code. Use the following command to execute the script:
   sudo python3 main.py [DO NOT RUN WITHOUT SUDO]

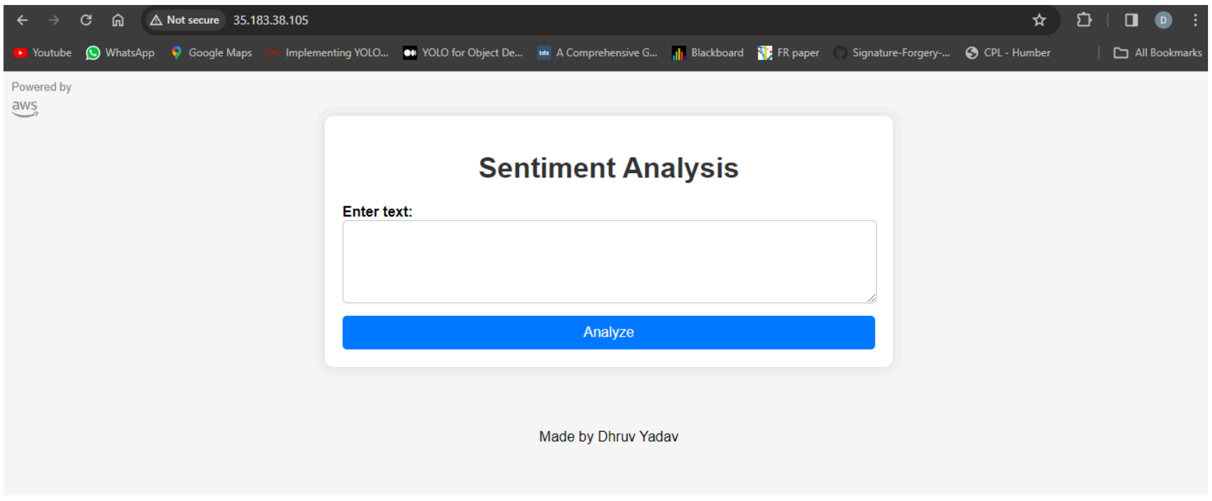
2. A Server Live string will confirm that the server has started running on port 80.



```
[ec2-user@ip-172-31-149-160 project1]$ sudo python3 main.py
====SERVER LIVE====
```

Final Website

The final website can be accessed by the URL http://35.183.38.105/ .
The code is also available at Github
https://github.com/BUSY-LOOPING/SentimentAnalysis-AWS .

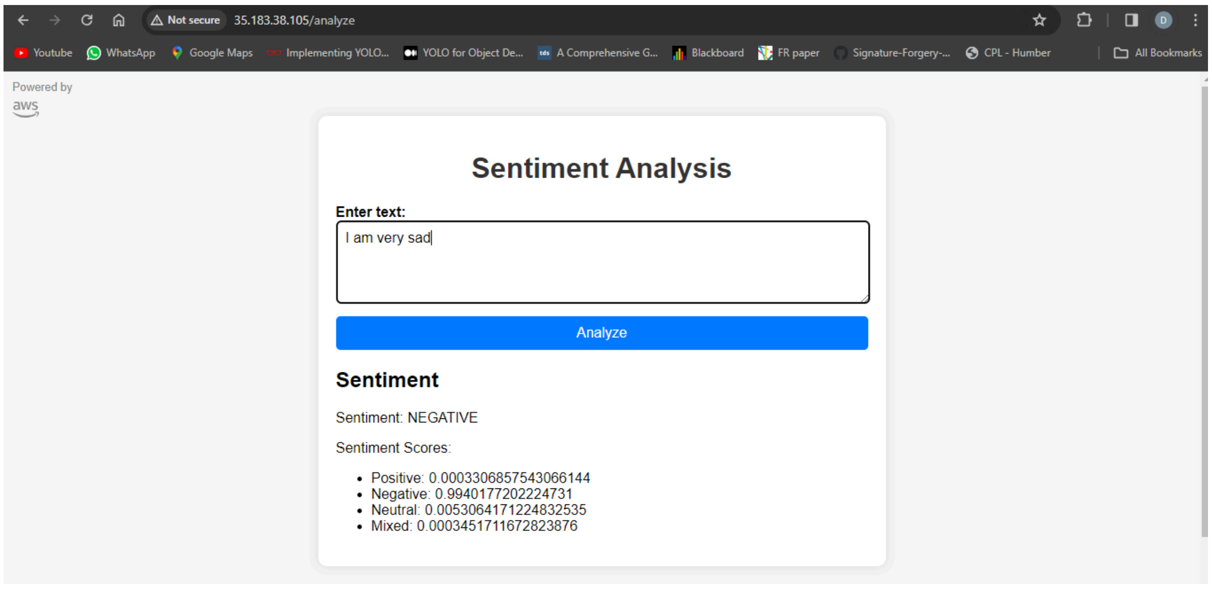


Sentiment Analysis

Enter text:

Analyze

Made by Dhruv Yadav



Sentiment Analysis

Enter text:

I am very sad

Analyze

Sentiment

Sentiment: NEGATIVE

Sentiment Scores:

- Positive: 0.0003306857543066144
- Negative: 0.9940177202224731
- Neutral: 0.0053064171224832535
- Mixed: 0.0003451711672823876