Ronjon Kundu
23215183

Setting up Python Environment
By using the code to make the environment for language detection

pip3 install iso-639

After that modify a python script to detect the language with the confidence level for four different languages

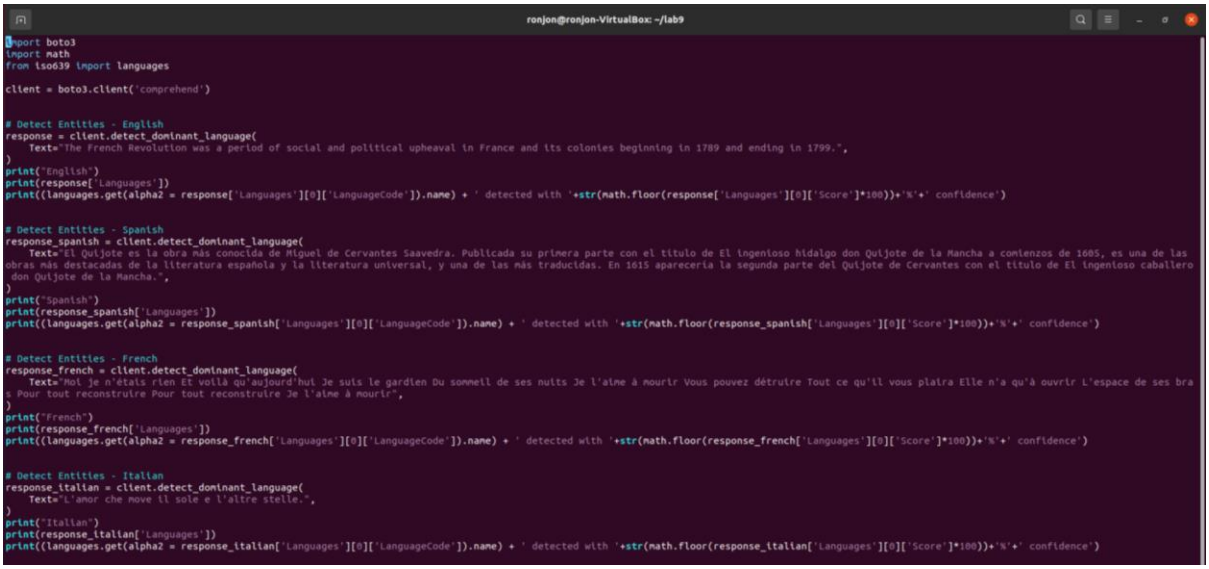


Fig 1: Detect languages code

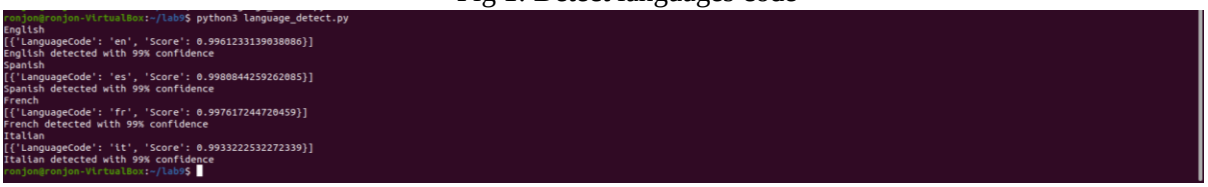


Fig 2: Output of language detection

After modifying the python script to detect the Sentiment of the language for four different languages

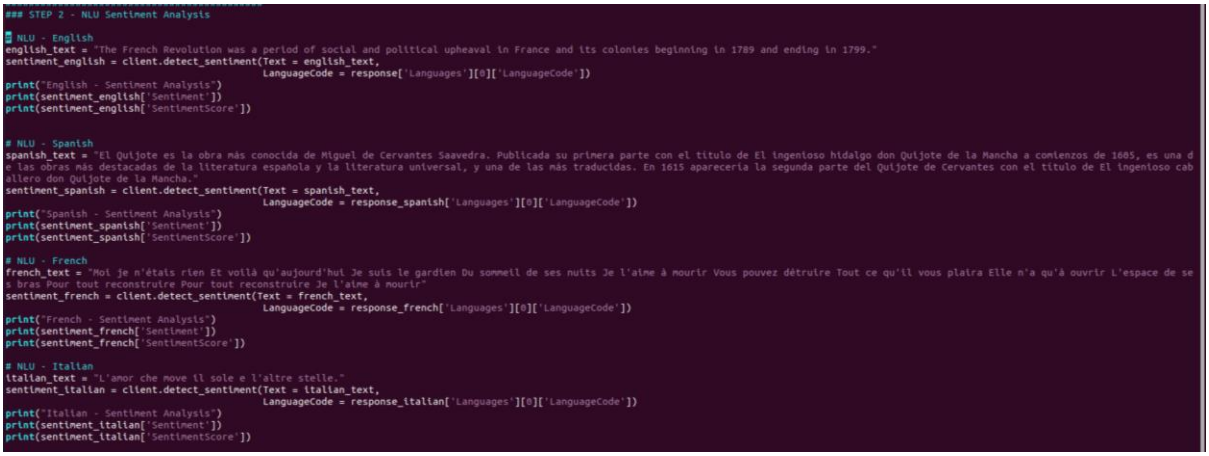


Fig 3: Detect languages sentiments code

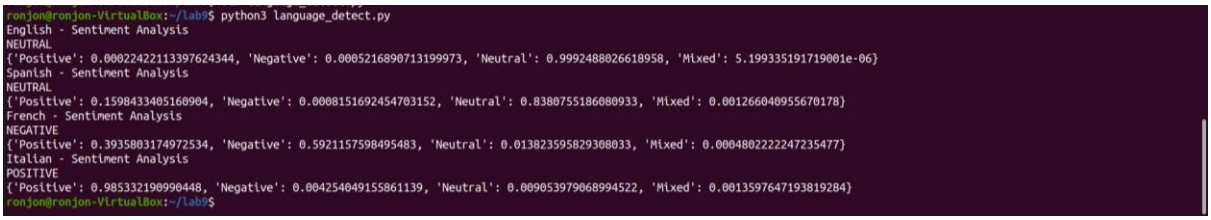


Fig 4: Detect languages sentiments output

After modifying the python script to detect the entities of the language for four different languages

```
### STEP 3 -Detecting entities

# Detect entities - English
entity_english = client.detect_entities(
    Text=english_text,
    LanguageCode=response['Languages'][0]['LanguageCode']
)
print("English")
print(entity_english['Entities'])

# Detect entities - Spanish
entity_spanish= client.detect_entities(
    Text=spanish_text,
    LanguageCode=response_spanish['Languages'][0]['LanguageCode']
)
print("Spanish")
print(entity_spanish['Entities'])

# Detect entities - French
entity_french = client.detect_entities(
    Text=french_text,
    LanguageCode=response_french['Languages'][0]['LanguageCode']
)
print("French")
print(entity_french['Entities'])

# Detect entities - Italian
entity_italian = client.detect_entities(
    Text=italian_text,
    LanguageCode=response_italian['Languages'][0]['LanguageCode']
)
print("Italian")
print(entity_italian['Entities'])
```

Fig 5: Detect languages entities code

Fig 6: Detect languages entities output

After modifying the python script to detect the keyphrases of the language for four different languages

```
### STEP 4 -Detecting keyphrases

# Detect keyphrases - English
keyphrase_english = client.detect_key_phrases(
    Text= english_text,
    LanguageCode = response['Languages'][0]['LanguageCode']
)
print("English")
print(keyphrase_english['KeyPhrases'])

# Detect keyphrases - Spanish
keyphrase_spanish = client.detect_key_phrases(
    Text= spanish_text,
    LanguageCode = response_spanish['Languages'][0]['LanguageCode']
)
print("Spanish")
print(keyphrase_spanish['KeyPhrases'])

# Detect keyphrases - French
keyphrase_french = client.detect_key_phrases(
    Text= french_text,
    LanguageCode = response_french['Languages'][0]['LanguageCode']
)
print("French")
print(keyphrase_french['KeyPhrases'])

# Detect keyphrases - Italian
keyphrase_italian = client.detect_key_phrases(
    Text= italian_text,
    LanguageCode = response_italian['Languages'][0]['LanguageCode']
)
print("Italian")
print(keyphrase_italian['KeyPhrases'])
```

Fig 7: Detect languages keyphrases code

Fig 8: Detect languages keyphrases output

After modifying the python script to detect the syntax of the language for four different languages

```
### STEP 5 -Detecting syntax

# Detect syntax - English
syntax_english = client.detect_syntax(
    Text=english_text,
    LanguageCode=response['Languages'][0]['LanguageCode']
)
print("English")
print(syntax_english['SyntaxTokens'])

# Detect syntax - Spanish
syntax_spanish = client.detect_syntax(
    Text=spanish_text,
    LanguageCode=response_spanish['Languages'][0]['LanguageCode']
)
print("Spanish")
print(syntax_spanish['SyntaxTokens'])

# Detect syntax - French
syntax_french = client.detect_syntax(
    Text=french_text,
    LanguageCode=response_french['Languages'][0]['LanguageCode']
)
print("French")
print(syntax_french['SyntaxTokens'])

# Detect syntax - Italian
syntax_italian = client.detect_syntax(
    Text=italian_text,
    LanguageCode=response_italian['Languages'][0]['LanguageCode']
)
print("Italian")
print(syntax_italian['SyntaxTokens'])
```

Fig 9: Detect languages syntax code

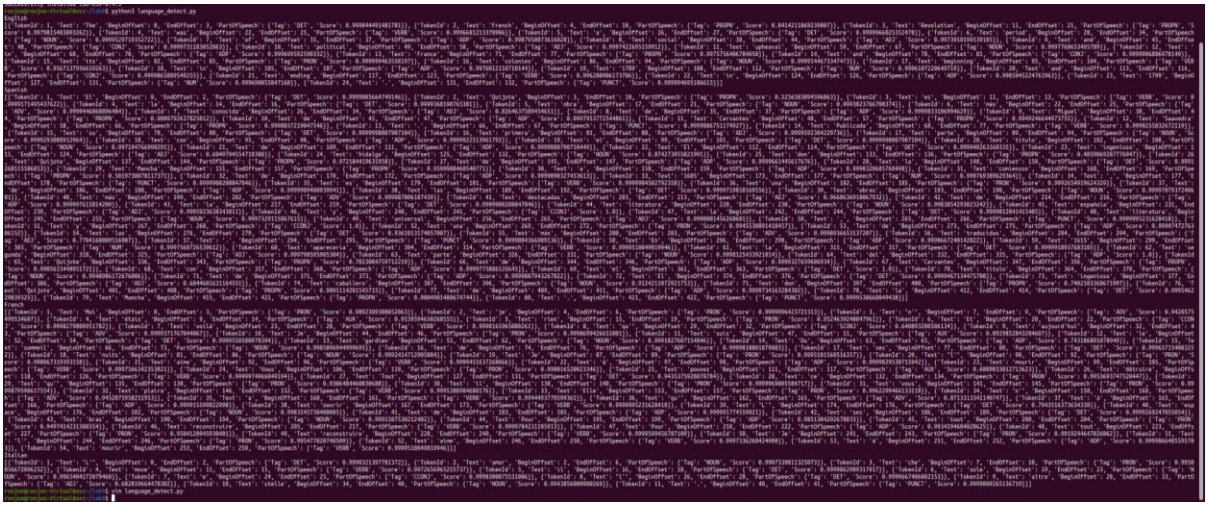


Fig 10: Detect languages syntax output

Entities - An *entity* of a text is the real-world object such as people, places, and commercial items in it.

Key Phrase - A *key phrase* is a noun the modifiers that distinguish the entities in a text.

Syntax – It is the words from the document. The nouns, verbs, adjectives and so on in a document. Syntax analysis helps to understand the relationship of the words in the document.

A bucket is created with AWS console and uploaded four different pictures for Label Recognition, Image Moderation, Facial Analysis and Text detection from an image.

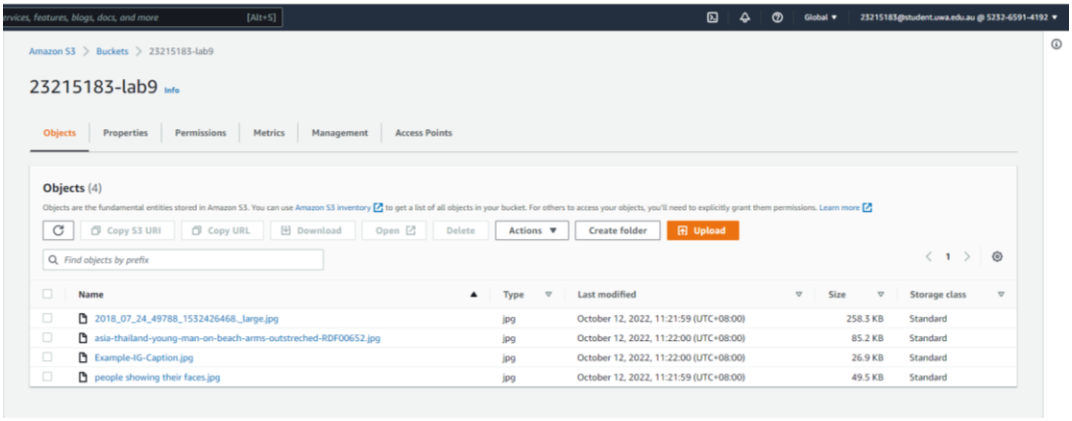


Fig 11: S3 bucket with pictures

Then modify a python script for Label Recognition

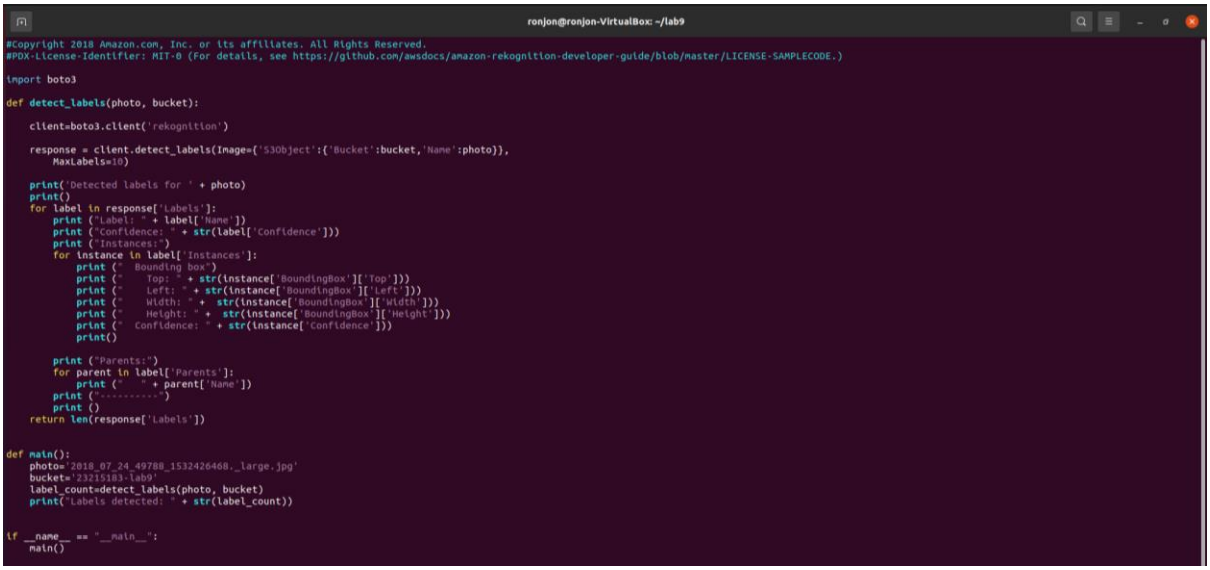
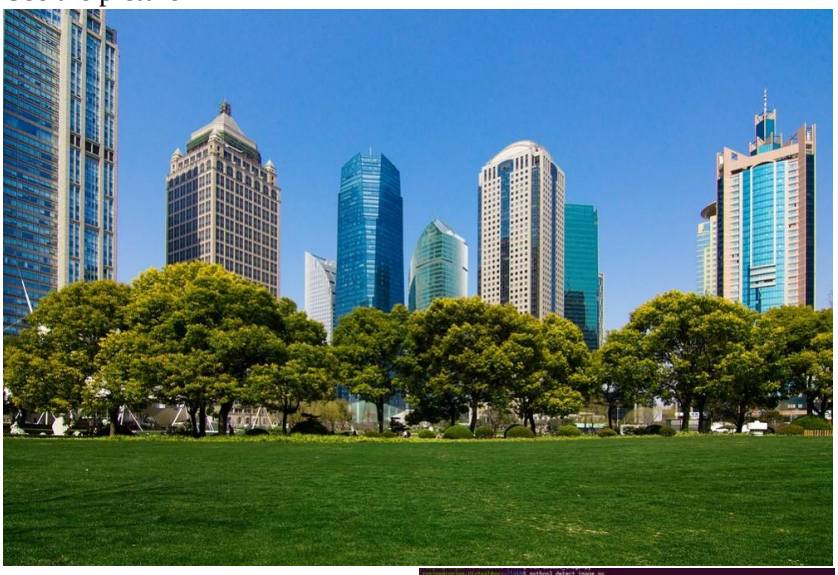


Fig 12: Code for label recognition

Use the picture



```
Detected Labels for 2018_07_24_00708_1532426468__large.jpg

Label: Grass
Confidence: 99.95637440632812
Instances:
Parents:
  Plant
----------

Label: Plant
Confidence: 99.95637440632812
Instances:
Parents:
----------

Label: City
Confidence: 98.78321838378906
Instances:
Parents:
  Urban
  Building
----------

Label: Urban
Confidence: 98.78321838378906
Instances:
Parents:
----------

Label: Building
Confidence: 98.78321838378906
Instances:
Parents:
----------

Label: High Rise
Confidence: 97.90077470027344
Instances:
Parents:
  City
  Urban
  Building
----------

Label: Lawn
Confidence: 94.08861210261719
Instances:
Parents:
  Grass
  Plant
----------

Label: Downtown
Confidence: 93.8046875
Instances:
Parents:
  City
  Urban
  Building
----------

Label: Park
Confidence: 88.25534820556664
Instances:
Parents:
  Lawn
  Grass
  Plant
----------

Label: Metropolis
Confidence: 81.72341156005586
```

Fig 13: output for label recognition

After modifying a python script for image moderation

ronjon@ronjon-VirtualBox: ~/lab9

```
import boto3

def moderate_image(photo, bucket):

    client=boto3.client('rekognition')

    response = client.detect_moderation_labels(Image={'S3Object':{'Bucket':bucket,'Name':photo}}
)

    print('Detected labels for ' + photo)
    for label in response['ModerationLabels']:
        print (label['Name'] + ' : ' + str(label['Confidence']))
        print (label['ParentName'])
    return len(response['ModerationLabels'])



def main():
    photo='asia-thailand-young-man-on-beach-arms-outstreched-RDF00652.jpg'
    bucket='23215183-lab9'
    label_count=moderate_image(photo, bucket)
    print("Labels detected: " + str(label_count))


if __name__ == "__main__":
    main()
```

Fig 14: Code for Image moderation

Using the picture

```
Detected labels for asia-thailand-young-man-on-beach-arms-outstreched-RDF00652.jpg
Suggestive : 86.2708969116211

Barechested Male : 86.2708969116211
Suggestive
Labels detected: 2
ronjon@ronjon-VirtualBox:~/lab9$
```

Fig 15: Output of image moderation

For facial analysis a python script is modified

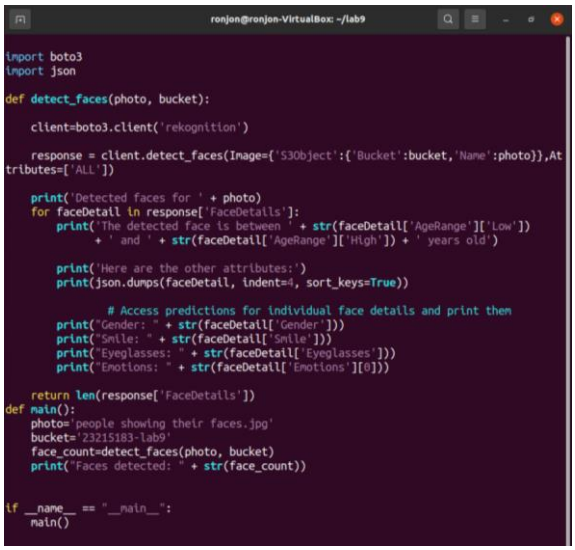


Fig 16: Code and picture used for facial analysis

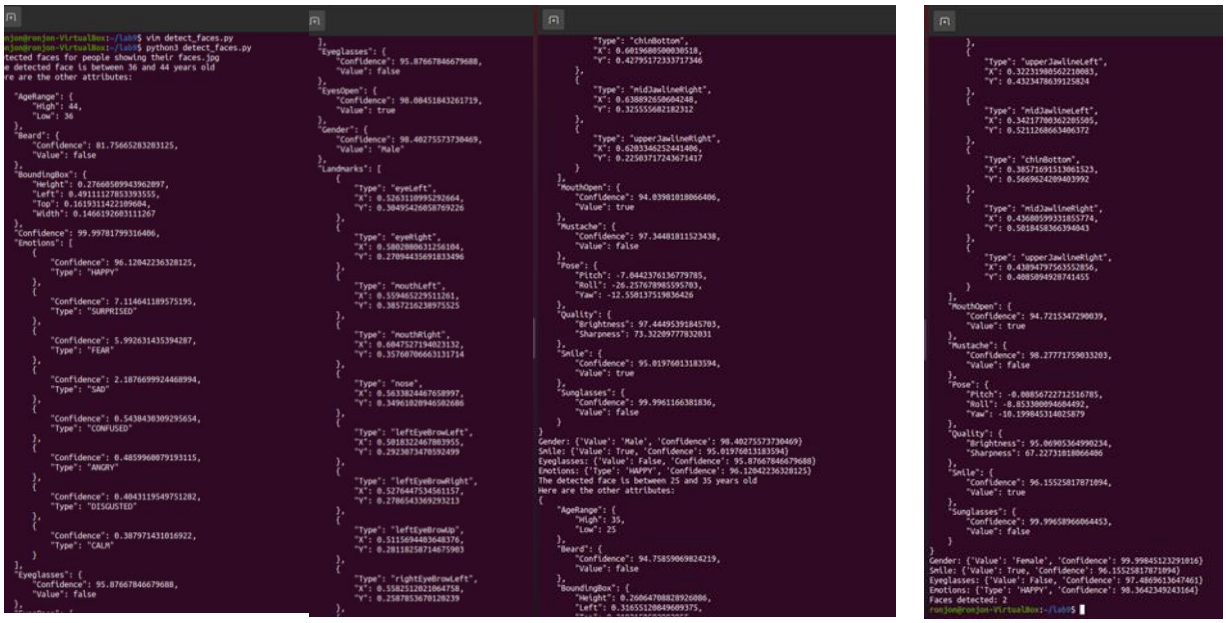


Fig 17: output for facial analysis

To extract text from an image modify a python script

ronjon@ronjon-VirtualBox: ~/lab9

```
#Copyright 2018 Amazon.com, Inc. or its affiliates. All Rights Reserved.
#PDX-License-Identifier: MIT-0 (For details, see https://github.com/awsdocs/amazon-rekogni
eveloper-guide/blob/master/LICENSE-SAMPLECODE.)

import boto3

def detect_text(photo, bucket):

    client=boto3.client('rekognition')

    response=client.detect_text(Image={'S3Object':{'Bucket':bucket,'Name':photo}})

    textDetections=response['TextDetections']
    print ('Detected text\n----------')
    for text in textDetections:
            print ('Detected text:' + text['DetectedText'])
            print ('Confidence: ' + "{:.2f}".format(text['Confidence']) + "%")
            print ('Id: {}'.format(text['Id']))
            if 'ParentId' in text:
                print ('Parent Id: {}'.format(text['ParentId']))
            print ('Type:' + text['Type'])
            print()
    return len(textDetections)

def main():

    bucket='23215183-lab9'
    photo='Example-IG-Caption.jpg'
    text_count=detect_text(photo,bucket)
    print("Text detected: " + str(text_count))


if __name__ == "__main__":
    main()
```

Fig 18: Code and picture used to extract text from an image

```
Detected text
----------
Detected text:COME SAIL AWAY
Confidence: 99.83%
Id: 0
Type:LINE

Detected text:COME
Confidence: 99.78%
Id: 1
Parent Id: 0
Type:WORD

Detected text:SAIL
Confidence: 100.00%
Id: 2
Parent Id: 0
Type:WORD

Detected text:AWAY
Confidence: 99.69%
Id: 3
Parent Id: 0
Type:WORD

Text detected: 4
ronjon@ronjon-VirtualBox:~/lab9$
```

Fig 19: output of extract text from an image

After completing all tasks in the lab delete the bucket

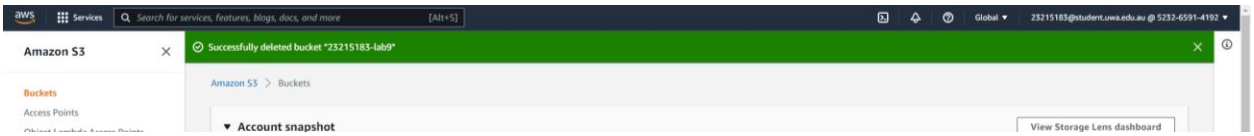


Fig 20: Delete bucket