Assignment-3

1.Create a bucket in IBM object storage.

```
from flask import Flask,redirect,url_for,render_template,request
import ibm_boto3
from ibm_botocore.client import Config, ClientError

COS_ENDPOINT="https://s3.jp-tok.cloud-object-storage.appdomain.cloud"
COS_API_KEY_ID=" "
COS_INSTANCE_CRN=""



# Create resource https://s3.ap.cloud-object-storage.appdomain.cloud
cos = ibm_boto3.resource("s3",
    ibm_api_key_id=COS_API_KEY_ID,
    ibm_service_instance_id=COS_INSTANCE_CRN,
    config=Config(signature_version="oauth"),
    endpoint_url=COS_ENDPOINT
)

app=Flask(__name__)



def get_item(bucket_name, item_name):
    print("Retrieving item from bucket: {0}, key: {1}".format(bucket_name, item_name))
    try:
        file = cos.Object(bucket_name, item_name).get()

        print("File Contents: {0}".format(file["Body"].read()))
    except ClientError as be:
        print("CLIENT ERROR: {0}\n".format(be))
    except Exception as e:
        print("Unable to retrieve file contents: {0}".format(e))


def get_bucket_contents(bucket_name):
    print("Retrieving bucket contents from: {0}".format(bucket_name))
    try:
        files = cos.Bucket(bucket_name).objects.all()
        files_names = []
        for file in files:
            files_names.append(file.key)
            print("Item: {0} ({1} bytes).".format(file.key, file.size))
        return files_names
    except ClientError as be:
        print("CLIENT ERROR: {0}\n".format(be))
    except Exception as e:
        print("Unable to retrieve bucket contents: {0}".format(e))


def delete_item(bucket_name, object_name):
    try:
        cos.delete_object(Bucket=bucket_name, Key=object_name)
        print("Item: {0} deleted!\n".format(object_name))
    except ClientError as be:
        print("CLIENT ERROR: {0}\n".format(be))
    except Exception as e:
        print("Unable to delete object: {0}".format(e))
def multi_part_upload(bucket_name, item_name, file_path):
    try:
        print("Starting file transfer for {0} to bucket: {1}\n".format(item_name, bucket_name))
        # set 5 MB chunks
        part_size = 1024 * 1024 * 5

        # set threadhold to 15 MB
        file_threshold = 1024 * 1024 * 15

        # set the transfer threshold and chunk size
        transfer_config = ibm_boto3.s3.transfer.TransferConfig(
            multipart_threshold=file_threshold,
            multipart_chunksize=part_size
        )

        # the upload_fileobj method will automatically execute a multi-part upload
        # in 5 MB chunks for all files over 15 MB
        with open(file_path, "rb") as file_data:
            cos.Object(bucket_name, item_name).upload_fileobj(
                Fileobj=file_data,
                Config=transfer_config
            )

        print("Transfer for {0} Complete!\n".format(item_name))
    except ClientError as be:
        print("CLIENT ERROR: {0}\n".format(be))
    except Exception as e:
        print("Unable to complete multi-part upload: {0}".format(e))


@app.route('/')
def index():
    files = get_bucket_contents('flaskapp123')
    return render_template('index.html', files = files)

@app.route('/deletefile', methods = ['GET', 'POST'])
def deletefile():
   if request.method == 'POST':
       bucket=request.form['bucket']
       name_file=request.form['filename']

       delete_item(bucket,name_file)
       return 'file deleted successfully'

   if request.method == 'GET':
       return render_template('delete.html')

@app.route('/uploader', methods = ['GET', 'POST'])
def upload():
   if request.method == 'POST':
       bucket=request.form['bucket']
       name_file=request.form['filename']
       f = request.files['file']
       multi_part_upload(bucket,name_file,f.filename)
       return 'file uploaded successfully <a href="/">GO to Home</a>'


   if request.method == 'GET':
       return render_template('upload.html')

if __name__=='__main__':
    app.run(host='0.0.0.0',port=8080,debug=True)
```

2.Upload an 5 images to IBM object storage and make it public.write HTML code to displaying all the 5 images.

```
<!DOCTYPE html>
<html>
<body>
<link rel="stylesheet" href="jeevii.css">
<h2>images</h2>
<img src="images.jpg" alt="Mountain" style="width:300px"><br>
<br>
<img src="images.jpg" alt="Mountain" style="width:300px"><br>
<br>
<img src="images.jpg" alt="Mountain" style="width:300px"><br>
<br>
<img src="images.jpg" alt="Mountain" style="width:300px"><br>
<br>
<img src="images.jpg" alt="Mountain" style="width:300px"><br>
<br>

</body>
</html>
```

3.Upload a CSS page to the object storage and use the same page in your HTML code.

```
body {
background:blue;
}
li {
    font-style: italic;
    color: red;
}
```

4.Design a chatbot using IBM watson assistant for hospital.Ex:user comes with query to know the branches for that hospital in your city.submit the web URL of that chat bot as a assignment.

```
# Import "chatbot" from
# chatterbot package.
from chatterbot import ChatBot

# Inorder to train our bot, we have
# to import a trainer package
# "ChatterBotCorpusTrainer"
from chatterbot.trainers import ChatterBotCorpusTrainer


# Give a name to the chatbot “corona bot”
# and assign a trainer component.
chatbot=ChatBot('corona bot')

# Create a new trainer for the chatbot
trainer = ChatterBotCorpusTrainer(chatbot)

# Now let us train our bot with multiple corpus
trainer.train("chatterbot.corpus.english.greetings",
              "chatterbot.corpus.english.conversations" )

response = chatbot.get_response('What is your Number')
print(response)
response = chatbot.get_response('Who are you?')
print(response)
```

5.Create watson assistant service with 10 steps and use 3 conditions in it.load that script in HTML page.

```
<html>
    <body>
  <a href="/">HOME</a>
 <a href="/uploader">Upload </a>
 <a href="/deletefile">Delete </a>
 <br><hr>

 <h1>IBM Upload File</h1>

       <form action = "/uploader" method = "POST"
          enctype = "multipart/form-data">
                 <input type = "text" placeholder="Enter bucket name" name = "bucket" />
                 <br>
                 <br>
                 <input type = "text" placeholder="Enter file name" name = "filename" />
                 <br>
                 <br>
          <input type = "file" name = "file" />
                 <br>
                 <br>
```

```
      <input type = "submit"/>
    </form>
  </body>
</html>
```