

Assignment Date	3 OCTOBER 2022
Student Name	RAVITHA R
Student Roll Name	721219104040
Maximum Marks	2Marks

```
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="erYOo-lq7dwN0LigQFZUw9ZhSjUSZPeUrZl8ValOZwoC"
```

```
COS_INSTANCE_CRN="crn:v1:bluemix:public:cloud-object-  
storage:global:a/11d1eefda09948f1a3075e4fab8e60f6:712106a7-5039-4a5e-8c83-c72c3638249a::"
```

```
# 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('flaskbucket')
```

```
    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)
```

Upload.html

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
    <meta charset="UTF-8">
```

```
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <title>Upload Files</title>
```

```
    <link rel="stylesheet" href="{{url_for('static',filename='css/style.css')}}">
```

```
</head>
```

```
<body>
```

```
<div class="section">
```

```
  <div class="menu">
```

```
    <ul class="menu-list">
```

```
      <li><a href="{{url_for('index')}}">Index</a></li>
```

```
      <li><a href="{{url_for('upload')}}">Upload</a></li>
```

```
      <li><a href="{{url_for('deletefile')}}">Delete</a></li>
```

```
    </ul>
```

```
  </div>
```

```
<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 class="file-inp" type="file" name="file" />
```

```
  <br>
```

```
  <br>
```

```
  <input class="btn-submit" type="submit" />
```

```
</form>
```

```
</div>
```

</body>

</html>

Index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Home</title>

<link rel="stylesheet" href="{{url_for('static',filename='css/style.css')}}">

</head>

<body>

<div class="section">

<div class="menu">

<ul class="menu-list">

Index

Upload

Delete

</div>


```
<hr>

    <h1>IBM Object Storage</h1>

    <h2>{% for row in files %}</h2>

    <div class="content">

        <h3>Filename : {{row}} </h3>

        

    </div>

    {% endfor %}

</div>

</body>

</html>
```

Delete.html

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Delete File</title>

    <link rel="stylesheet" href="{{url_for('static',filename='css/style.css')}}">

</head>
```

```
<body>

  <div class="section">

    <div class="menu">

      <ul class="menu-list">

        <li><a href="{{url_for('index')}}">Index</a></li>

        <li><a href="{{url_for('upload')}}">Upload</a></li>

        <li><a href="{{url_for('deletefile')}}">Delete</a></li>

      </ul>

    </div>

    <hr>

    <h1>IBM Object Storage</h1>

    <form action="/deletefile" method="POST">

      <input type="text" placeholder="Enter bucket name" name="bucket" />

      <br>

      <br>

      <input type="text" placeholder="Enter file name" name="filename" />

      <br>

      <br>

      <input class="btn-submit" type="submit" />

    </form>

  </div>
```

```
</body>
```

```
</html>
```

Style.css

```
* {  
  
  margin: 0;  
  
  padding: 0;  
  
  color: white;  
  
  font-family: Arial, Helvetica, sans-serif;  
}
```

```
html {  
  
  width: 100%;  
  
  height: 100vh;  
}
```

```
body {  
  
  width: 100%;  
  
  height: 100%;  
  
  background-color: rgb(8, 82, 124);  
  
  display: flex;  
  
  justify-content: center;  
}
```

```
.section {  
  
  width: 30%;  
  
  height: fit-content;  
  
  display: flex;  
  
  flex-direction: column;  
  
  /* border: 1px solid #fff; */  
  
  
  box-shadow: 2px 2px 1px 2px rgba(167, 167, 167, 0.644);  
  
  border-radius: 2rem;  
  
  border-left: 1px solid rgba(255, 255, 255, 0.3);  
  border-top: 1px solid rgba(255, 255, 255, 0.3);  
  
  align-items: center;  
  
  padding-bottom: 5rem;  
  
  margin-top: 5rem;  
}
```

```
a {  
  
  text-decoration: none;  
}
```

```
.menu {  
  
  margin-top: 2rem;  
  
  width: 100%;  
  
  height: 3rem;
```

```
display: flex;

align-items: center;

/* background-color: bisque; */

justify-content: center;

border-bottom: 2px solid rgba(255, 255, 255, 0.555);

padding-bottom: 2rem;

}

.menu-list {

width: 100%;

list-style-type: none;

text-transform: uppercase;

display: flex;

}

.menu-list li {

width: 100%;

display: flex;

justify-content: space-around;

}

a {

box-shadow: 1px 1px 2px 2px rgba(167, 167, 167, 0.4);
```

```
padding: 1rem 1.4rem;  
border-radius: 1rem;  
}
```

```
h1 {  
  /* padding-left: 2rem; */  
  margin-top: 1rem;  
}
```

```
h2 {  
  margin-top: 1rem;  
}
```

```
.content {  
  /* margin-left: 2rem; */  
  margin-top: 2rem;  
}
```

```
form {  
  /* margin-left: 2rem; */  
  margin-top: 1rem;  
}
```

```
input {  
  height: 2.6rem;  
  width: 18rem;
```

```
padding-left: .6rem;  
border: none;  
border-radius: .4rem;  
color: #000;  
}
```

```
.btn-submit {  
  width: 5rem;  
  padding: .2rem;  
  height: 2rem;  
  color: rgb(104, 104, 104);  
  text-transform: uppercase;  
}
```

```
.file-inp {  
  width: 15rem;  
  height: 2rem;  
  border: none;  
}
```

```
.btn-submit:hover {  
  color: #000;  
}
```

