```
import ibm_boto3
from ibm_botocore.client import Config, ClientError
from ibm_s3transfer.aspera.manager import AsperaTransferManager
from ibm_s3transfer.aspera.manager import AsperaConfig
from flask import Flask, render_template, url_for, request, redirect

COS_ENDPOINT="https://s3.jp-tok.cloud-object-storage.appdomain.cloud"
COS_API_KEY_ID="wprEMAxyjHj5sPI959wL_3HJczOWRbYn52XUuLrDSJON"
COS_INSTANCE_CRN="crn:v1:bluemix:public:cloud-object-
storage:global:a/602bcedcf9224f7b8c2e1aed60258292:b846251f-3216-44c8-b123-4e13e3571cda::"

# 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
)

ms_transfer_config = AsperaConfig(multi_session=2, multi_session_threshold_mb=100)
transfer_manager = AsperaTransferManager(client=client, transfer_config=ms_transfer_config)

app=Flask(__name__)

bucket_name = "flask-application"
download_filename = "E:\IMS\static\css\Styles.css"
object_name = "Styles.css"

with AsperaTransferManager(client) as transfer_manager:
   future = transfer_manager.download(bucket_name, object_name, download_filename)
   future.result()
```

```
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 = []
        print(files)
        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))


@app.route('/')
def index():
    files = get_bucket_contents('flask-application')
```

```
    return render_template('index.html', files = files)


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