

Assignment-3

Date	10 October 2022
Name	Mohana sakthi.G.S
Team ID	PNT2022TMID30531
Project Name	Skill and Job Recommender Application

1. CREATE A BUCKET IN IBM OBJECT STORAGE.

The screenshot shows the IBM Cloud Object Storage console. The left sidebar contains navigation links: Cloud Object Storage, Storage instances, Cloud Object Storage-1e, Buckets, Integrations, Endpoints, Usage details, Service credentials, Connections, and Plan. The main content area shows the bucket 'skillandjobbot' with tabs for Objects, Configuration, and Permissions. The Objects tab is active, displaying a table of objects. The table has columns for Object name, Archived, Size, and Last modified. Five objects are listed: K1.jpg (160.1 KB, 2022-10-24 11:02 PM), K2.jpg (189.3 KB, 2022-10-24 11:02 PM), K3.jpg (188.2 KB, 2022-10-24 11:02 PM), K5.jpg (124.2 KB, 2022-10-24 11:02 PM), and K6.jpg (466.9 KB, 2022-10-24 11:03 PM). An 'Upload' button is visible in the top right corner of the object list area. Below the table, there is a message: 'Drag and drop files (objects) here or click to upload'.

Object name	Archived	Size	Last modified
K1.jpg		160.1 KB	2022-10-24 11:02 PM
K2.jpg		189.3 KB	2022-10-24 11:02 PM
K3.jpg		188.2 KB	2022-10-24 11:02 PM
K5.jpg		124.2 KB	2022-10-24 11:02 PM
K6.jpg		466.9 KB	2022-10-24 11:03 PM

**Upload an 5 images to ibm object storage and make it public.
Write html code todisplaying all the 5 images.**

IBM Cloud

Search resources and products...

Cloud Object Storage

Storage instances

Cloud Object Storage-le

Buckets

Integrations

Endpoints

Usage details

Service credentials

Connections

Plan

skillandjobbot

Transfers Details Actions...

Objects Configuration Permissions

If you're seeing more usage than expected, versions count towards your usage or you may have incomplete uploads [Learn more](#)

Prefix filter

Upload

Object name	Archived ⓘ	Size	Last modified
<input type="checkbox"/> K1.png		160.1 KB	2022-10-24 11:02 PM
<input type="checkbox"/> K2.png		189.3 KB	2022-10-24 11:02 PM
<input type="checkbox"/> K3.png		188.2 KB	2022-10-24 11:02 PM
<input type="checkbox"/> K5.png		124.2 KB	2022-10-24 11:02 PM
<input type="checkbox"/> K6.png		466.9 KB	2022-10-24 11:03 PM

Drag and drop files (objects) here or click to upload

IBM Cloud

Type here to search

ENG 11:03 PM IN 10/24/2022

IBM Cloud

Search resources and products...

Cloud Object Storage

Storage instances

Cloud Object Storage-le

Buckets

Integrations

Endpoints

Usage details

Service credentials

Connections

Plan

Bucket access policies

Manage access to this bucket by creating IAM policies for users and service IDs. Users and service IDs must also have an instance level viewer role (or higher) to use the console or to list buckets using the REST API.

Access policies

Public access

Access policy update

Access group policy created

A new access policy for this bucket was created for the group: Public Access

Status: Enabled

Role for this policy: Content Reader

As a Content Reader, one can read and list objects in the bucket.

Create access policy

Context-based restrictions

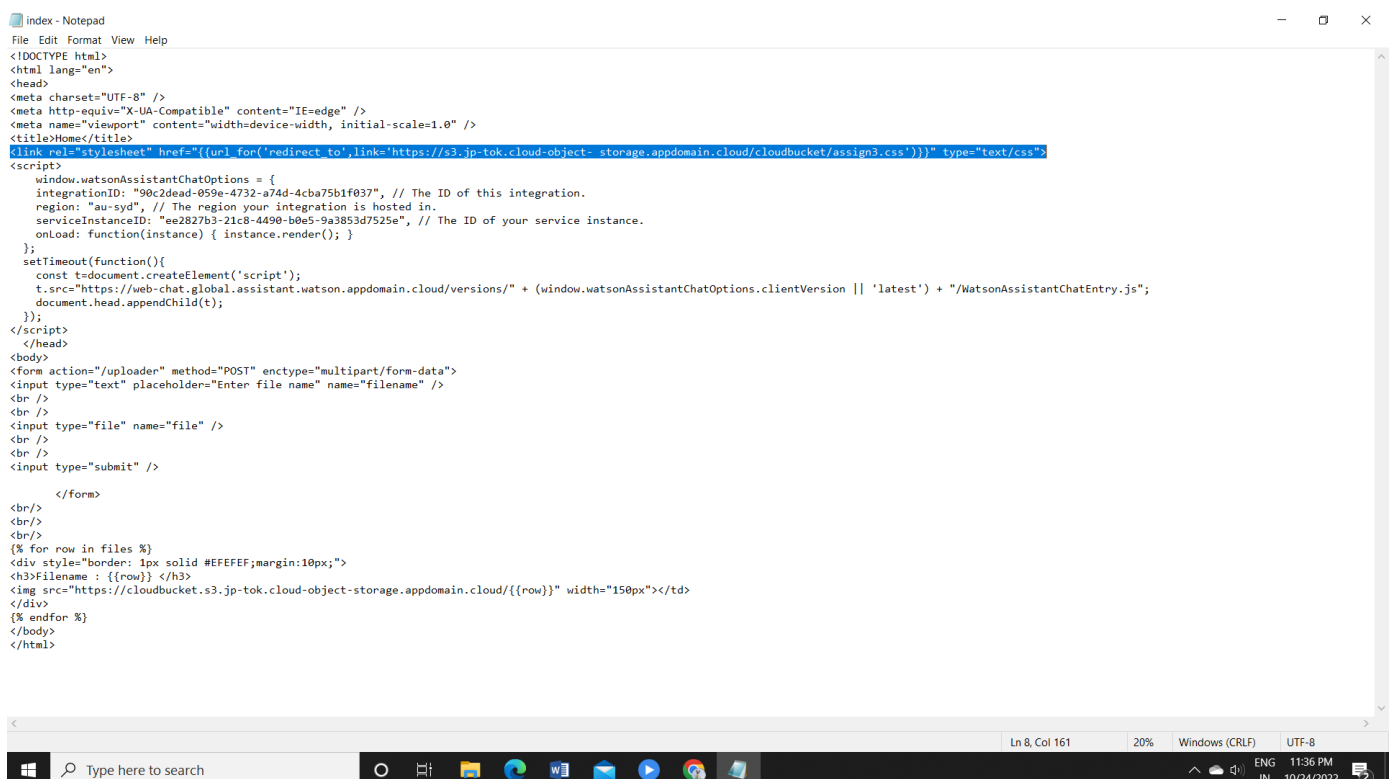
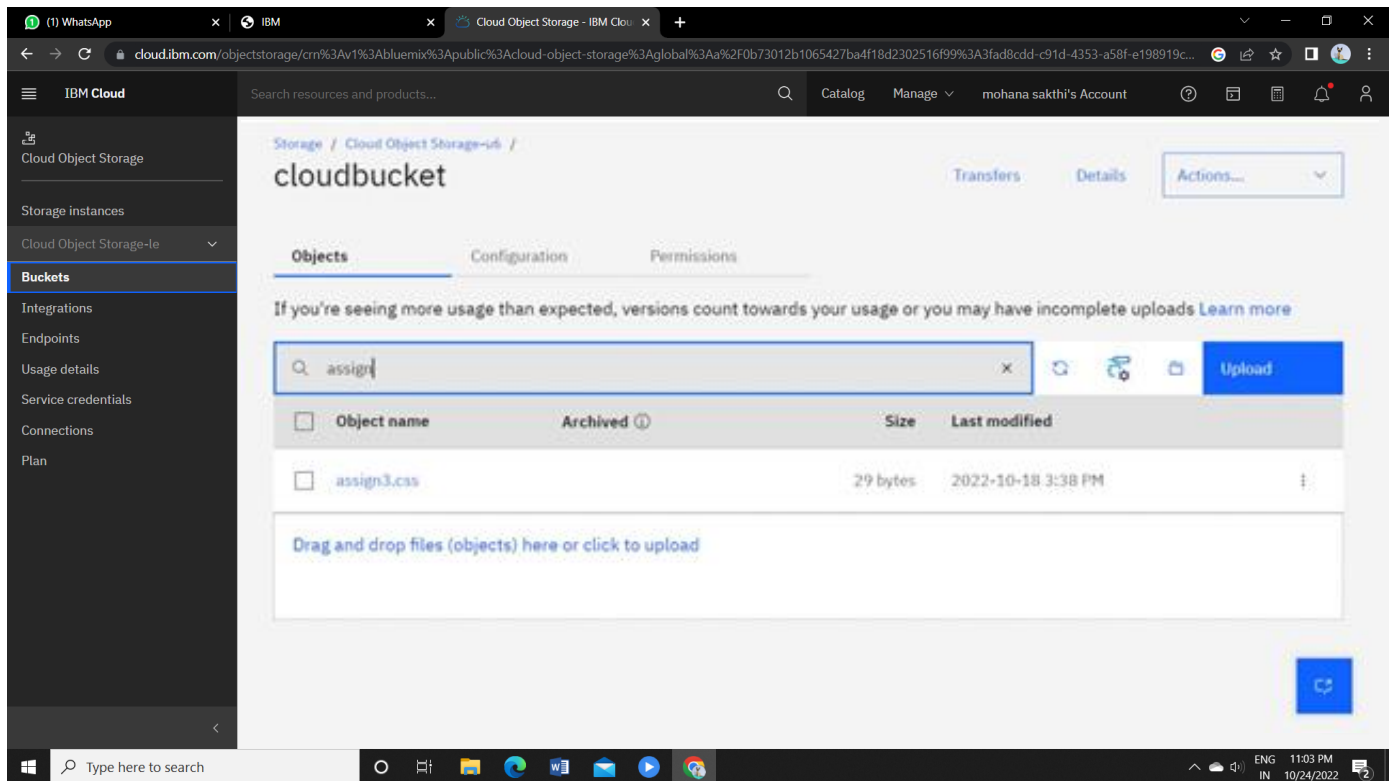
Firewall (legacy)

IBM Cloud

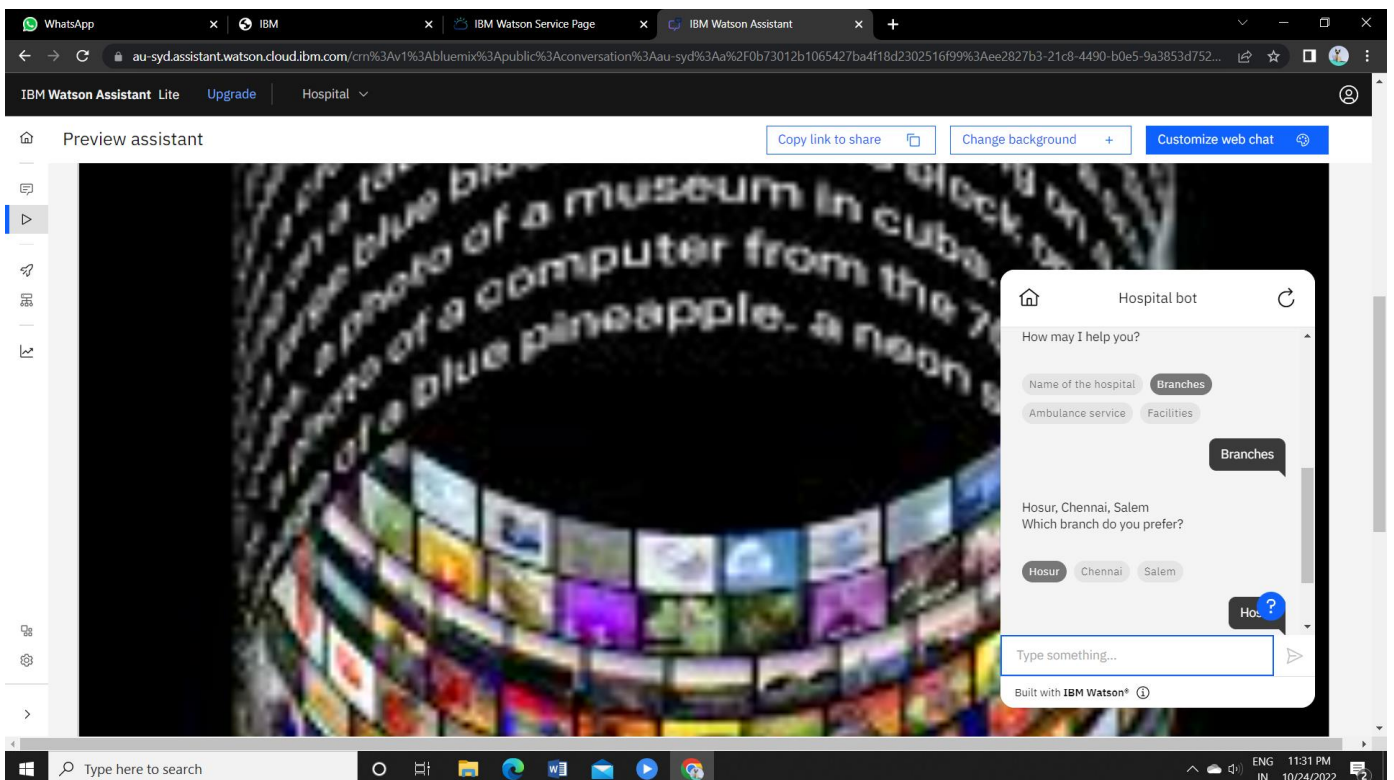
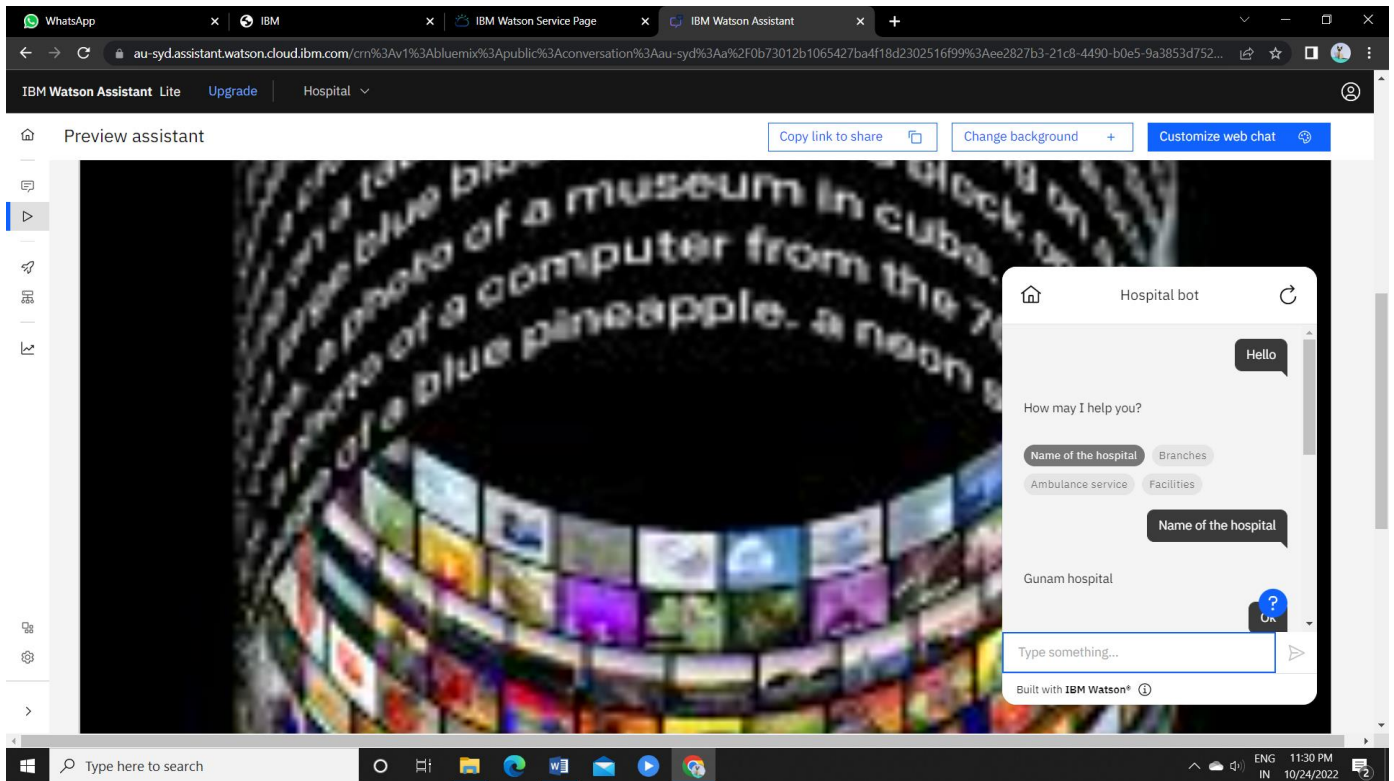
Type here to search

ENG 11:03 PM IN 10/24/2022

2. Upload a css page to the object storage and use the same page in your HTML code.



3. Design a chatbot using IBM Watson assistant for hospital.



Web URL for Assistant:

<https://web-chat.global.assistant.watson.appdomain.cloud/preview.html?backgroundImageURL=https%3A%2F%2Fau-syd.assistant.watson.cloud.ibm.com%2Fpublic%2Fimages%2Fupx-ee2827b3-21c8-4490-b0e5-9a3853d7525e%3A%3Aa631a952-bd36-4487-81f2-246f2a791a5b&integrationID=90c2dead-059e-4732-a74d-4cba75b1f037®ion=au-syd&serviceInstanceID=ee2827b3-21c8-4490-b0e5-9a3853d7525e>

4. Create Watson assistant service with 10 steps and use 3 conditions in it. Load that script in HTML page.

The screenshot displays the IBM Watson Assistant interface in a web browser. The top navigation bar includes 'IBM Watson Assistant Lite', 'Upgrade', and 'Hospital'. The main content area is divided into two panels. The left panel shows a conversation flow with 10 steps. Step 5 is 'Bed availability', Step 7 is 'Confirmation', Step 8 is 'Parking facilities', Step 9 is 'Which specialist do you prefer?', and Step 10 is 'Others'. Step 10 is currently selected, showing a 'Free text' input field and an 'Action complete' button. The right panel shows the configuration for Step 10, which is 'with conditions'. It lists one condition: 'If All of this is true: 9. Which specialist ... is Others'. Below the conditions, there is a 'New condition group' button. The 'Assistant says' section shows the text 'Kindly mention the specialist you need to visit.' and a 'Preview' button. The bottom of the interface shows a Windows taskbar with various application icons and a system clock indicating 11:23 PM on 10/24/2022.

Included 3 conditions in steps:

This screenshot shows the IBM Watson Assistant configuration interface for a chatbot named "Hospital". The left sidebar displays the conversation flow with three steps. Step 2 is highlighted, showing a user input "Name of the hospital" and a system response "Gunam hospital". The main panel shows the configuration for Step 2, which is triggered by the condition "1. How may I help y... is Name of the hospital". The assistant's response is "Gunam hospital".

Customer starts with: Hello

Conversation steps

1 How may I help you? (Name of the ... Branches +2)

2 1 is Name of the hospital (Gunam hospital) (Free text)

3 1 is Branches (Hosur, Chennai, Salem Which branch do you prefer?) (Hosur Chennai +1)

Step 2 is taken with conditions

Conditions 1 condition

If All of this is true:

1. How may I help y... is Name of the hospital

Assistant says

Gunam hospital

User enters free text

This screenshot shows the IBM Watson Assistant configuration interface for the same chatbot, now showing Step 3. The left sidebar shows the updated conversation flow. Step 3 is highlighted, showing a user input "Branches" and a system response "Hosur, Chennai, Salem Which branch do you prefer?". The main panel shows the configuration for Step 3, which is triggered by the condition "1. How may I help y... is Branches". The assistant's response is "Hosur, Chennai, Salem Which branch do you prefer?".

Customer starts with: Hello

Conversation steps

1 How may I help you? (Name of the ... Branches +2)

2 1 is Name of the hospital (Gunam hospital) (Free text)

3 1 is Branches (Hosur, Chennai, Salem Which branch do you prefer?) (Hosur Chennai +1)

4 Why do you prefer this location? (Free text)

Step 3 is taken with conditions

Conditions 1 condition

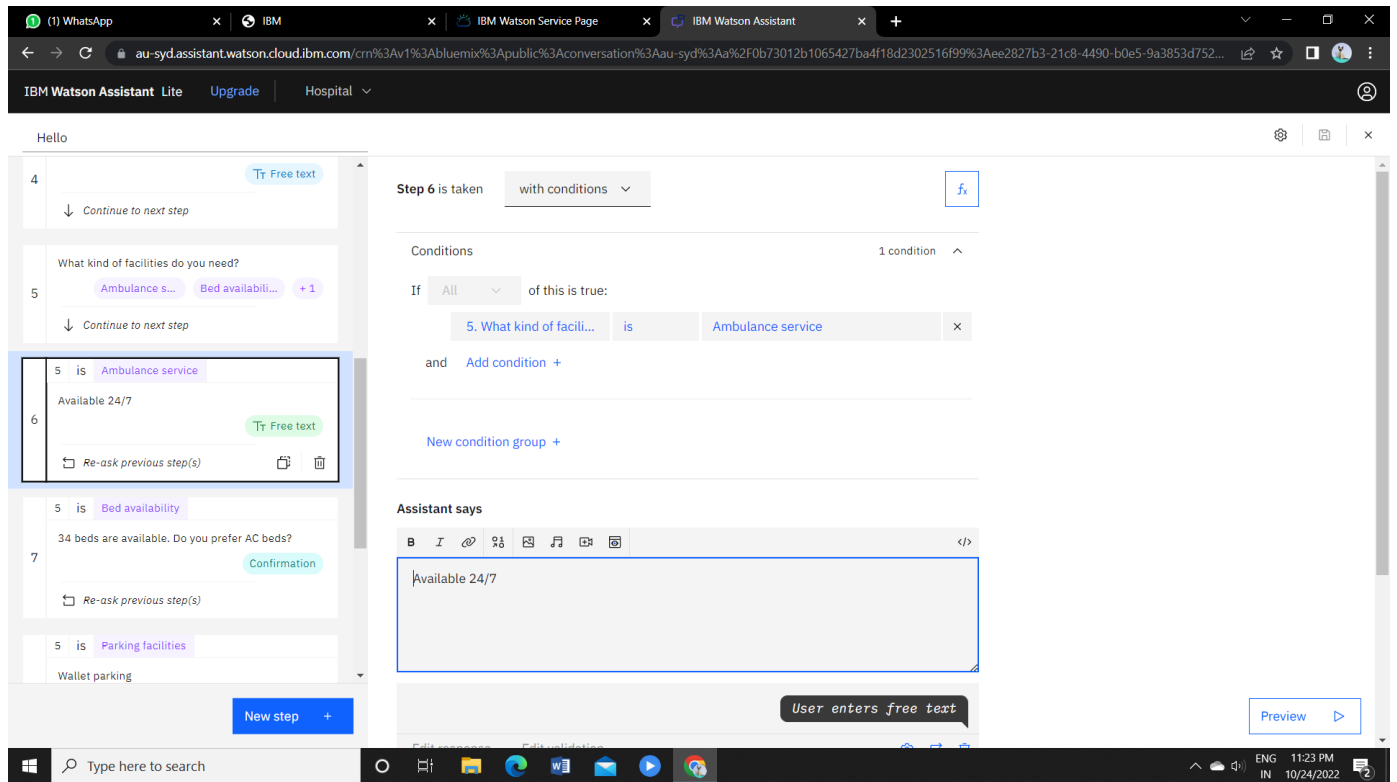
If All of this is true:

1. How may I help y... is Branches

Assistant says

Hosur, Chennai, Salem Which branch do you prefer?

Hosur Chennai Salem



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('redirect_to',link='https://s3.jp-tok.cloud-object-storage.appdomain.cloud/cloudbucket/assign3.css')} }" type="text/css">

    <script>
window.watsonAssistantChatOptions = {
  integrationID: "90c2dead-059e-4732-a74d-4cba75b1f037", // The ID of this integration.
  region: "au-syd", // The region your integration is hosted in.
  serviceInstanceID: "ee2827b3-21c8-4490-b0e5-9a3853d7525e", // The ID of your service instance.
  onLoad: function(instance) { instance.render(); }
};
setTimeout(function(){
  const t=document.createElement('script');
  t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion || 'latest') + "/WatsonAssistantChatEntry.js";
  document.head.appendChild(t);
});

    </script>

  </head>

  <body>

    <form action="/uploader" method="POST" enctype="multipart/form-data">

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

      <br />

      <br />

      <input type="file" name="file" />

      <br />

      <br />

      <input type="submit" />

    </form>

  </body>

</html>
```



```

</form>

<br/>

<br/>

<br/>

{% for row in files %}

    <div style="border: 1px solid #EFEFEF;margin:10px;">

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

        </td>

    </div>

{% endfor %}

</body>

</html>

```

App.py

```

import io

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=""


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__)

@app.route('/')
def index():
    try:
        files = cos.Bucket('cloudbucket').objects.all()
        files_names = []
        for file in files:
            files_names.append(file.key)
            print(file)
            print("Item: {0} ({1} bytes)".format(file.key, file.size))
        return render_template('index.html',files=files_names)

    except ClientError as be:
        print("CLIENT ERROR: {0}\n".format(be))
        return render_template('index.html')
    except Exception as e:
        print("Unable to retrieve bucket contents: {0}".format(e))
        return render_template('index.html')

@app.route('/uploader',methods=['POST'])
def upload():
    name_file=request.form['filename']
    f = request.files['file']
    try:
        part_size = 1024 * 1024 * 5

        file_threshold = 1024 * 1024 * 15

        transfer_config = ibm_boto3.s3.transfer.TransferConfig(
            multipart_threshold=file_threshold,

```

```
        multipart_chunksize=part_size
    )

    content = f.read()
    cos.Object('cloudbucket', name_file).upload_fileobj(
        Fileobj=io.BytesIO(content),
        Config=transfer_config
    )
    return redirect(url_for('index'))
```

```
except ClientError as be:
    print("CLIENT ERROR: {0}\n".format(be))
    return redirect(url_for('index'))
```

```
except Exception as e:
    print("Unable to complete multi-part upload: {0}".format(e))
    return redirect(url_for('index'))
```

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