

Assignment-3

Date	10 October 2022
Team ID	PNT2022TMID30665
Project Name	News Tracker Application.

1. CREATE A BUCKET IN IBM OBJECT STORAGE.

Cloud Object Storage - IBM Cloud

Search resources and products...

Buckets

Buckets serve as containers for objects, and can be individually configured in terms of their location, resiliency, billing rates, security, and object lifecycle rules.

Search

Create bucket +

Name	Public access ⓘ	Location ⓘ	Storage class	Created
ntbot	No	jp-tok	Smart Tier	2022-11-02 10:42 AM

26°C Cloudy

10:50 02-11-2022

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

Cloud Object Storage - IBM Cloud

Storage instances

Cloud Object Storage-m1

Buckets

Integrations

Endpoints

Usage details

Service credentials

Connections

Plan

Search resources and products...

ntbot

TransfersDetailsActions...

ObjectsConfigurationPermissions

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

Prefix filter

☐

Object name

Archived ⓘ

Size

Last modified

☐

L... pg

20.6 KB

2022-11-02 10:48 AM

☐

L... pg

26.1 KB

2022-11-02 10:49 AM

☐

L... pg

22.0 KB

2022-11-02 10:49 AM

☐

L... pg

10.3 KB

2022-11-02 10:49 AM

☐

L... pg

37.5 KB

2022-11-02 10:49 AM

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

Upload

26°C Cloudy

10:50 02-11-2022

Cloud Object Storage - IBM Cloud

Storage instances

Cloud Object Storage-m1

Buckets

Integrations

Endpoints

Usage details

Service credentials

Connections

Plan

Search resources and products...

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

Warning:

Access policy update

Access group policy created

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

To delete/edit go to the [IAM console](#).

Create access policy

Context-based restrictions

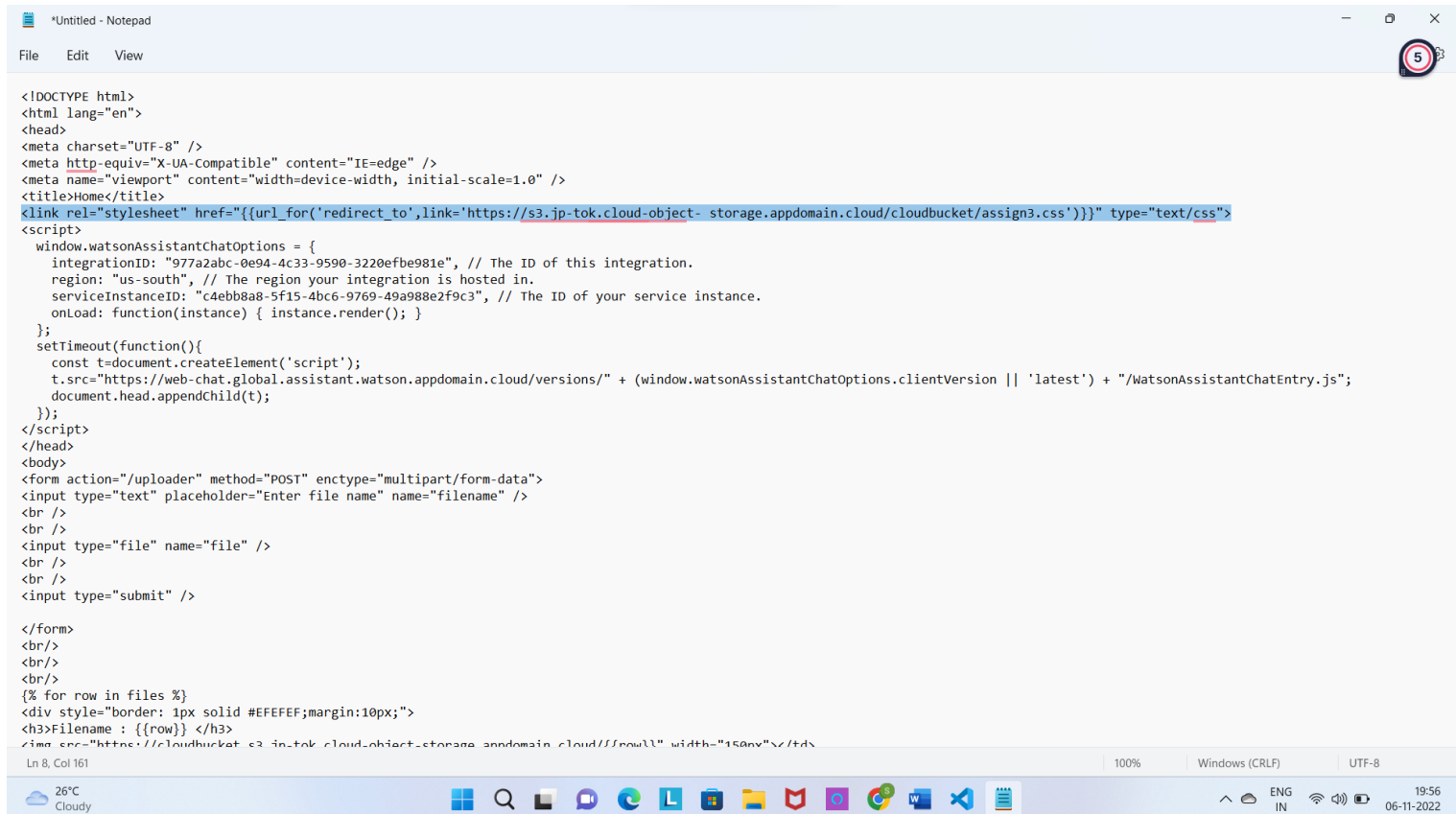
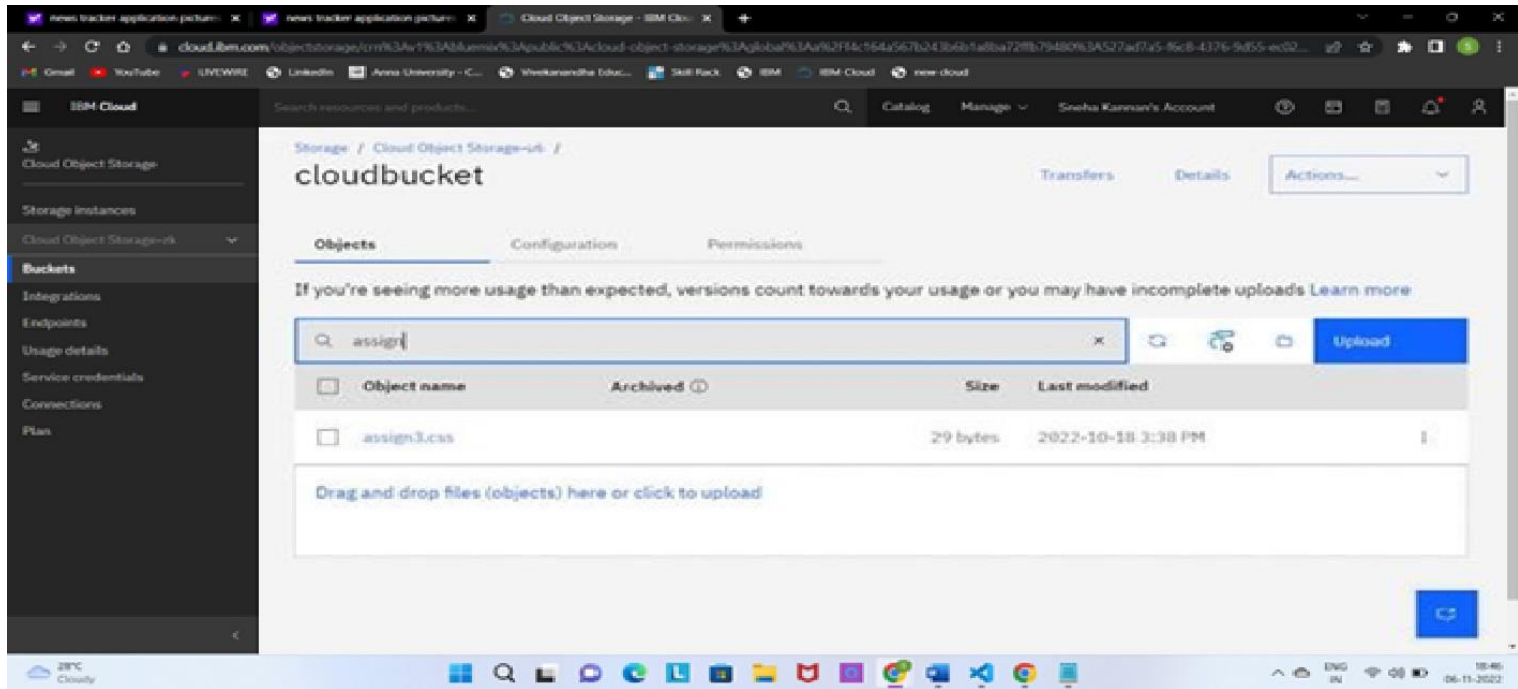
Firewall (legacy)

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

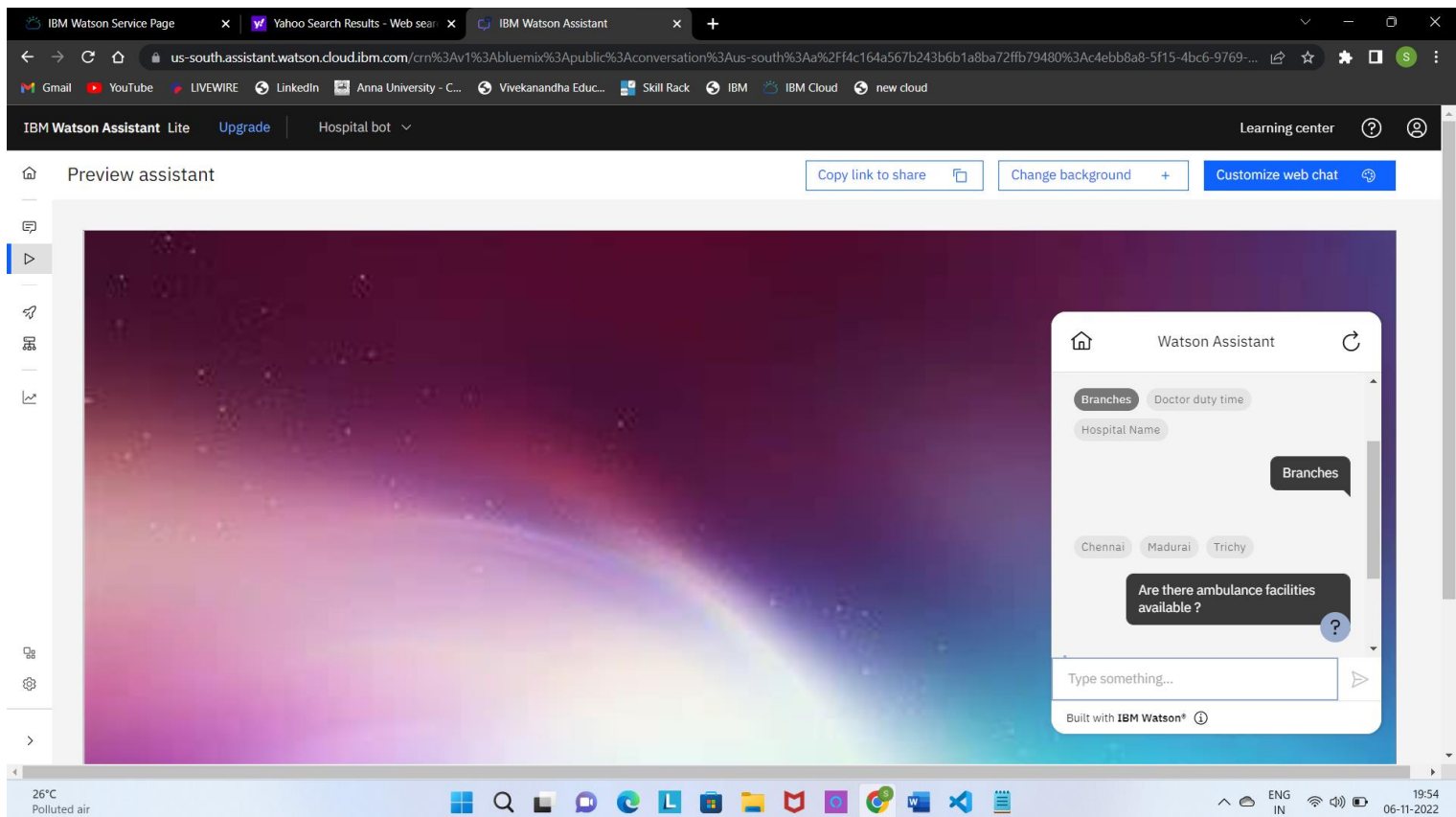
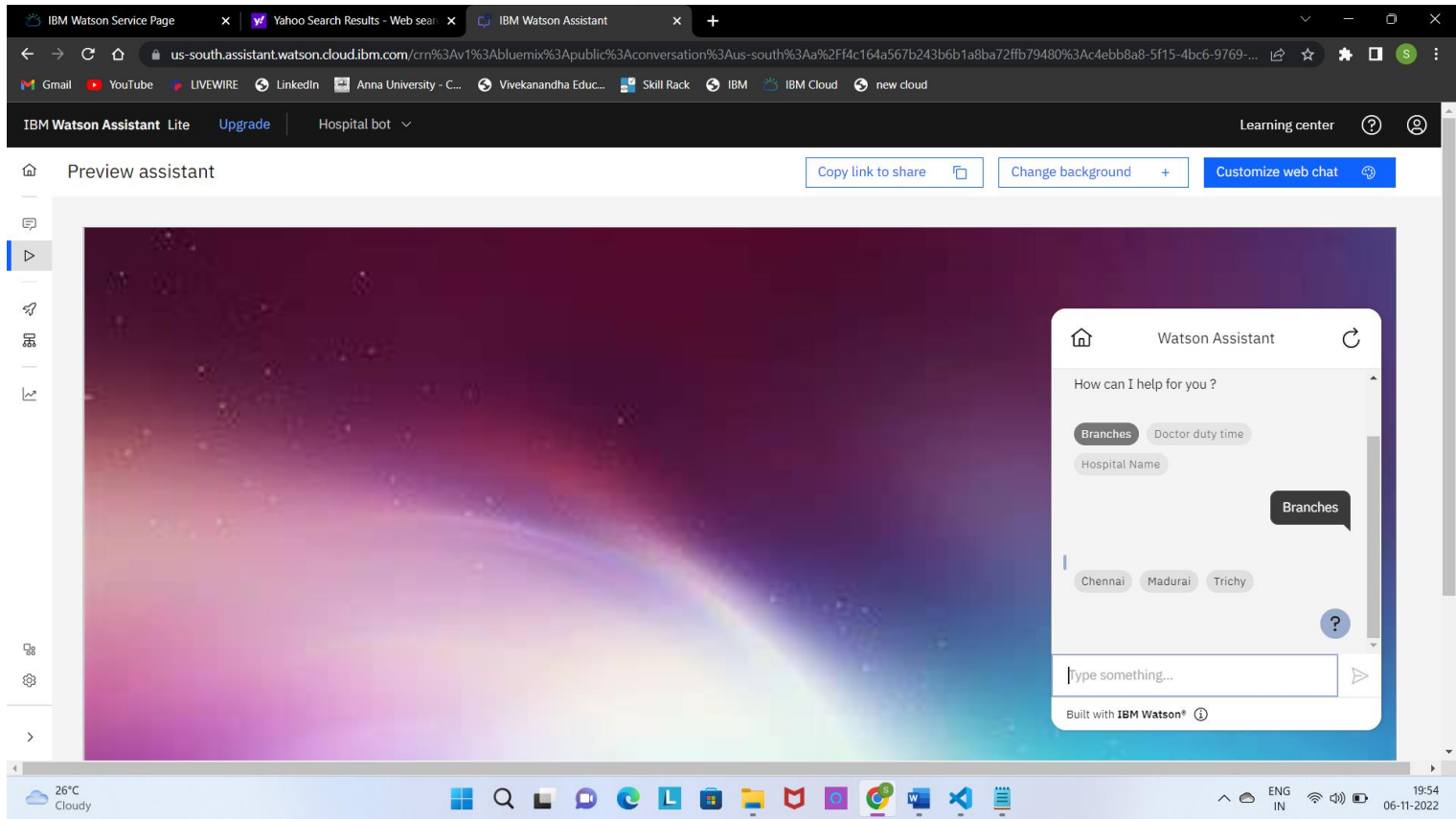
26°C Cloudy

10:51 02-11-2022

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%2Fus-south.assistant.watson.cloud.ibm.com%2Fpublic%2Fimages%2Fupx-c4ebb8a8-5f15-4bc6-9769-49a988e2f9c3%3A%3A4533f33c-f8a2-47b0-9463-ef7081baa701&integrationID=977a2abc-0e94-4c33-9590-3220efbe981e®ion=us-south&serviceInstanceID=c4ebb8a8-5f15-4bc6-9769-49a988e2f9c3>

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 console interface. The top navigation bar includes the IBM logo and various service links. The main content area shows a workflow editor with 10 steps. Step 10 is highlighted, showing a 'Helpline Number of a hospital?' prompt with a 'Number' input field and an 'Action complete' status. The 'Assistant says' section shows a 'Helpline Number of a hospital?' prompt. The 'And then' section shows an 'End the action' button. A 'Preview' window on the right shows a chat interface with a 'Greet customer' button and a 'Welcome, how can I assist you?' message. A 'Save your action' warning is visible at the bottom of the preview window.

Included 3 conditions in steps:

The screenshot shows the IBM Watson Assistant interface with a focus on Step 8. The left sidebar displays a sequence of steps: Step 7 (Male/Female), Step 8 (highlighted), Step 9 (Dr.sha, Dryam, +2), and Step 10 (Helpline Number of a hospital?). Step 8 is currently empty, with buttons for 'Dr.shan' and 'Dr.raaj'. The main panel shows the configuration for Step 8, which is set to 'with conditions'. A single condition is defined: 'If All of this is true: 7. Which doctor wo... is Male'. The 'Assistant says' section contains a text box with the placeholder 'For example: What type of transfer would you like to make?'. A 'Preview' button is visible in the bottom right corner.

The screenshot shows the IBM Watson Assistant interface with a focus on Step 7. The left sidebar displays a sequence of steps: Step 6 (Confirmation), Step 7 (highlighted), Step 8 (Dr.shan, Dr.raaj), and Step 9 (Dr.sha, Dryam, +2). Step 7 is currently empty, with buttons for 'Male' and 'Female'. The main panel shows the configuration for Step 7, which is set to 'without conditions'. The 'Assistant says' section contains a text box with the placeholder 'Which doctor would you need to consult?'. Below the text box are buttons for 'Male' and 'Female', and links for 'Edit response' and 'Edit validation'. The 'And then' section contains a dropdown menu with 'Continue to next step' selected. A 'Preview' button is visible in the bottom right corner.

IBM Watson Assistant LiteUpgradeHospital

us-south.assistant.watson.cloud.ibm.com/crn%3Av1%3Abluemix%3Aus-south-1%3Aassistant%3A...with conditions

IBM Watson Assistant LiteUpgradeHospital

Welcome

Continue to next step

1 is Specialist

This step has no content

2

ENT Pediatrician +2

Continue to next step

1 is Duty timing for doctor

10.00 am to 1.00 pm

3

Time

Continue to next step

1 is Bed Availability

25

4

Number

Continue to next step

1 is Visitors Timing

New step +

Step 3 is taken with conditions

Conditions

If All of this is true:

1. How can I help yo... is Duty timing for doctor

and Add condition +

New condition group +

Assistant says

10.00 am to 1.00 pm

Preview

User enters a time-based value

26°C
Raining now



ENG IN 12:43
02-11-2022

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: "977a2abc-0e94-4c33-9590-3220efbe981e", // The ID of this integration.
        region: "us-south", // The region your integration is hosted in.
        serviceInstanceID: "c4ebb8a8-5f15-4bc6-9769-49a988e2f9c3", // 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>
<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(files)
```

```
    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)
```