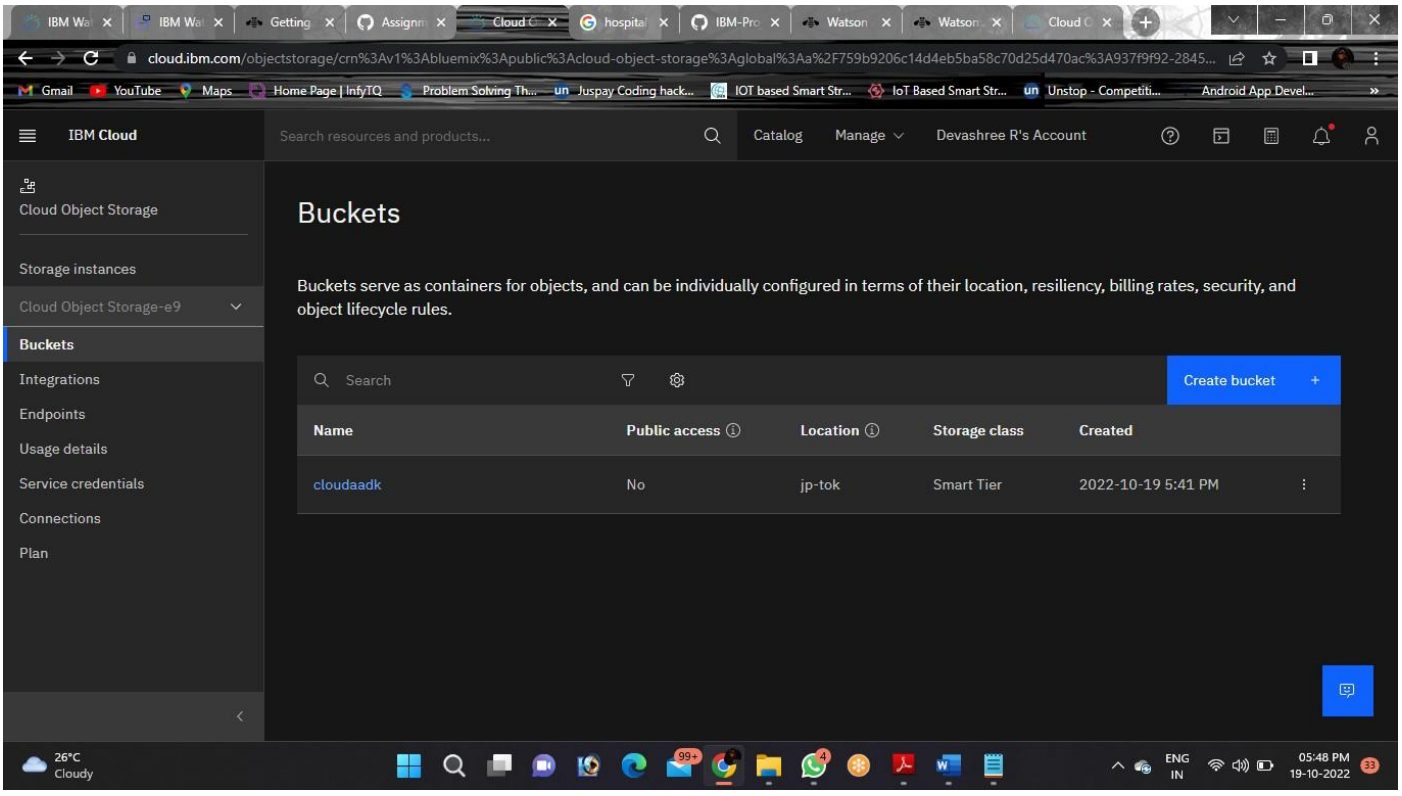
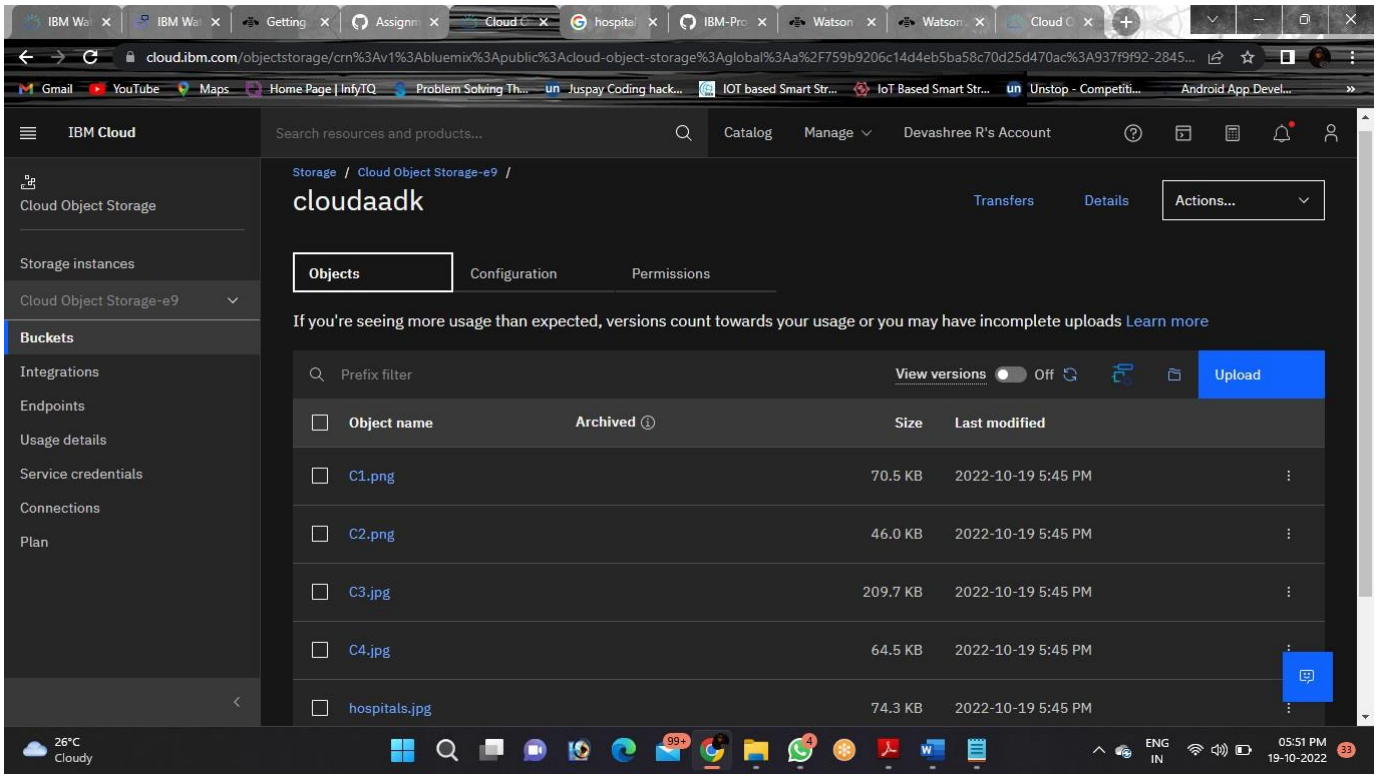
Assignment-3

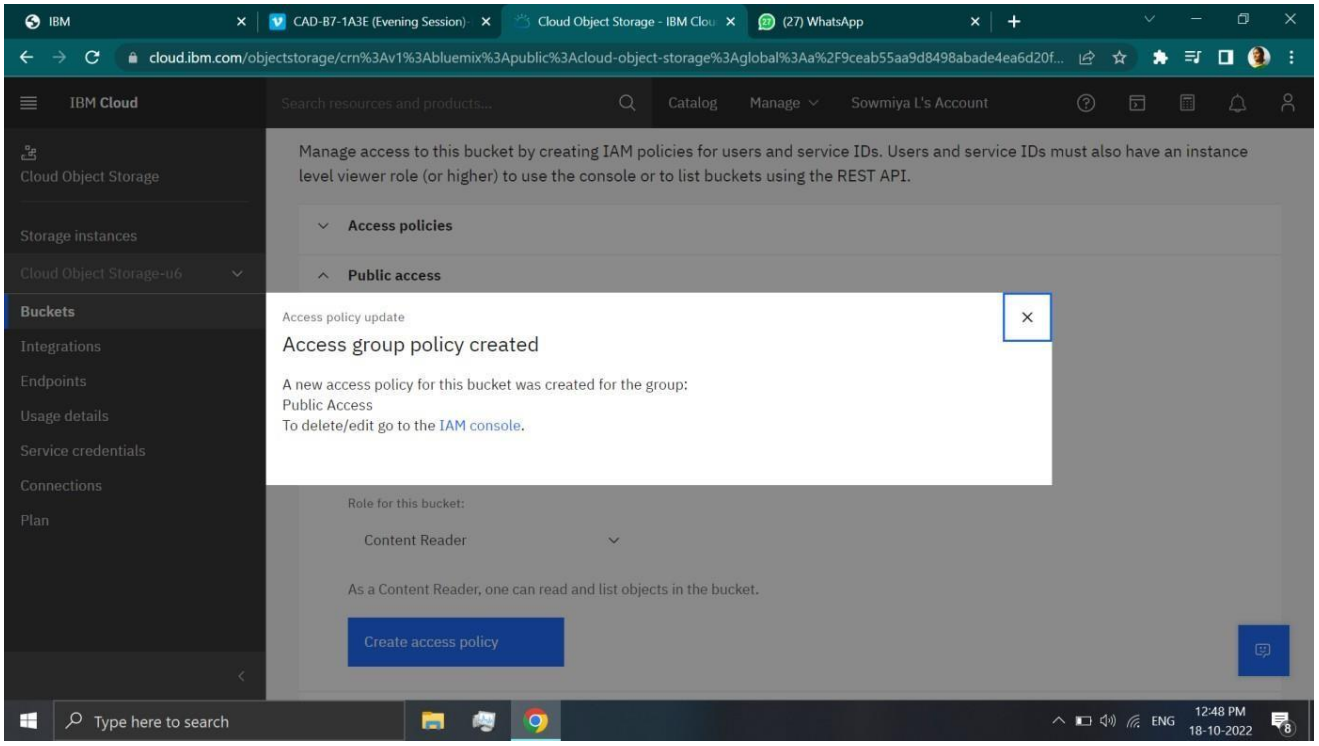
Date	10 October 2022
Team ID	PNT2022TMID15066
Project Name	PLASMA DONOR APPLICATION

1. CREATE A BUCKET IN IBM OBJECT STORAGE.

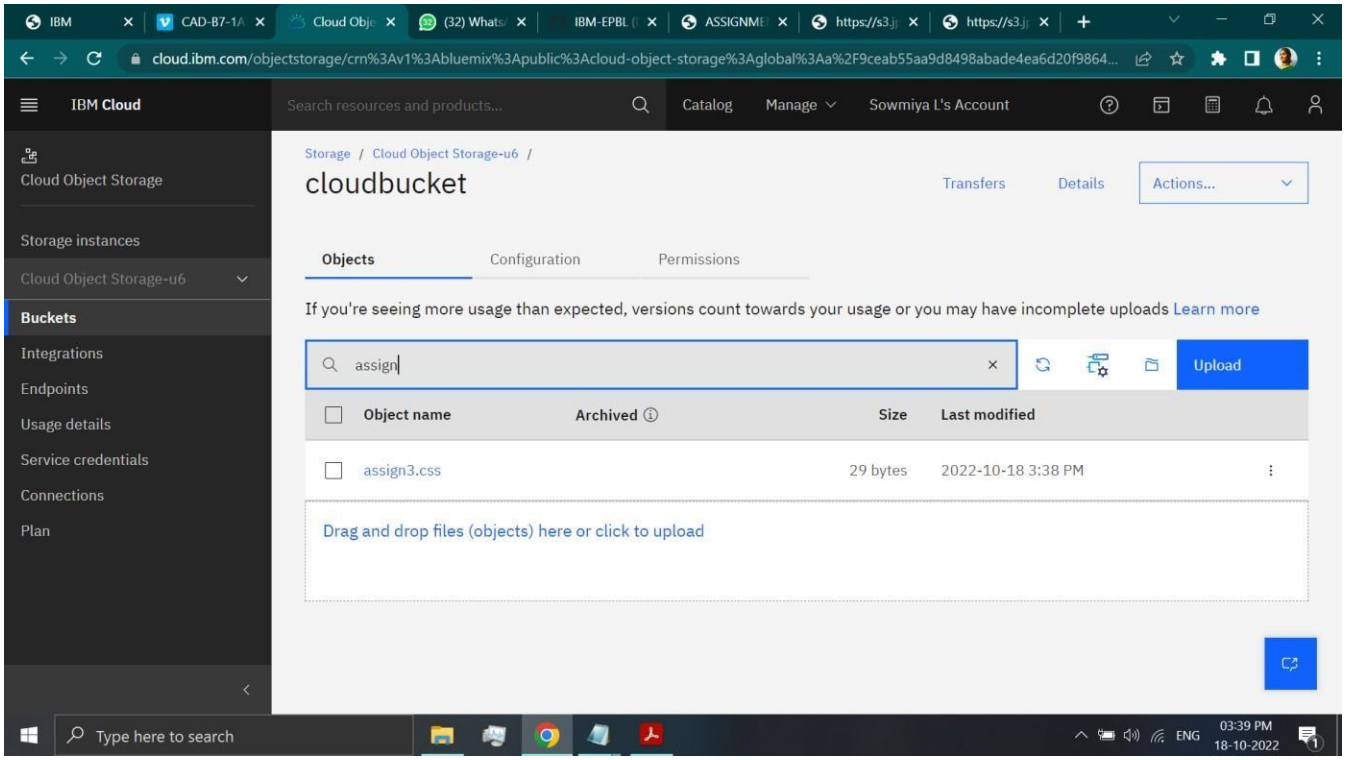


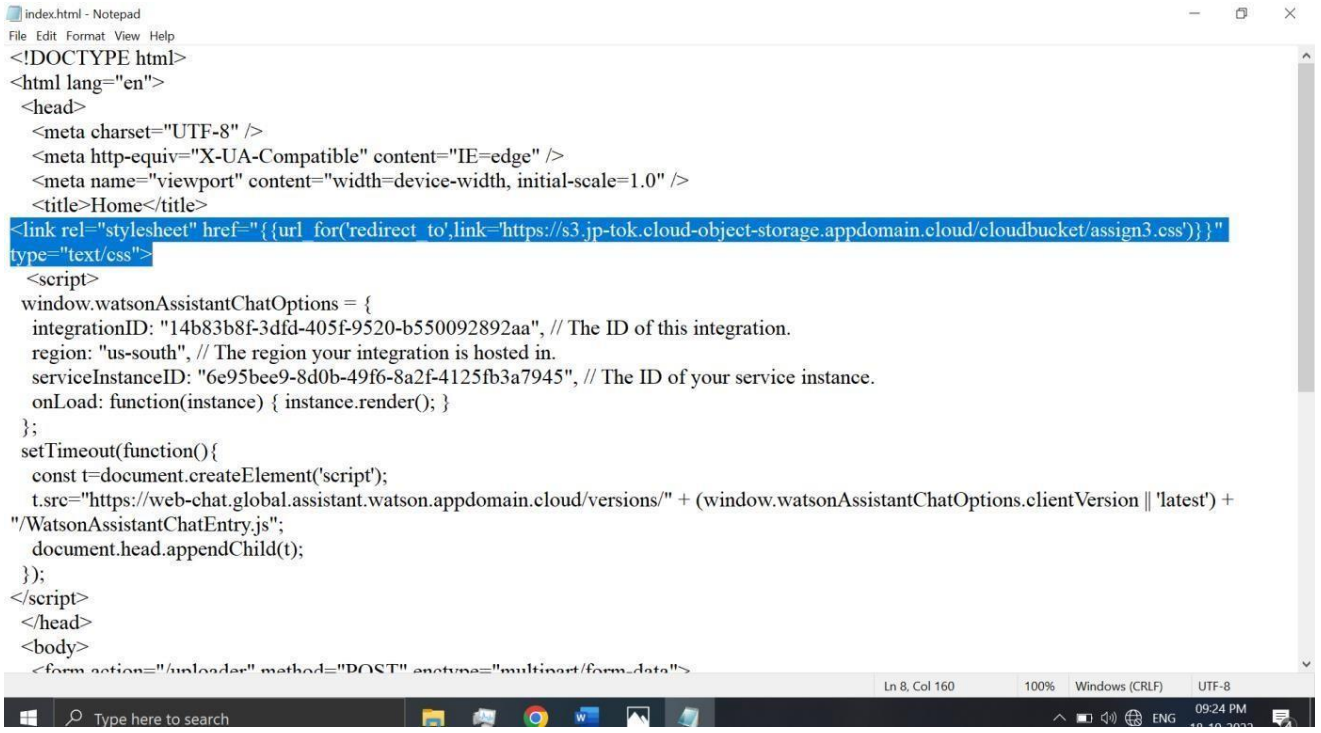
2. Upload an 5 images to ibm object storage and make it public. write html code todisplaying all the 5 images.


cloud.ibm.com/objectstorage/crn%3Av1%3Abluemix%3Apublic%3Acloud-object-storage%3Aglobal%3Aa%2F759b9206c14d4eb5ba58c70d25d470ac%3A937f9f92-2845...
IBM Cloud
Search resources and products...
Catalog
Manage
Devashree R's Account
Cloud Object Storage
Storage instances
Cloud Object Storage-e9
Buckets
Integrations
Endpoints
Usage details
Service credentials
Connections
Plan
Storage / Cloud Object Storage-e9 /
cloudaadk
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
View versions
Off
Upload
Object name
Archived
Size
Last modified
C1.png 70.5 KB 2022-10-19 5:45 PM
C2.png 46.0 KB 2022-10-19 5:45 PM
C3.jpg 209.7 KB 2022-10-19 5:45 PM
C4.jpg 64.5 KB 2022-10-19 5:45 PM
hospitals.jpg 74.3 KB 2022-10-19 5:45 PM
26°C
Cloudy
ENG
IN
05:51 PM
19-10-2022

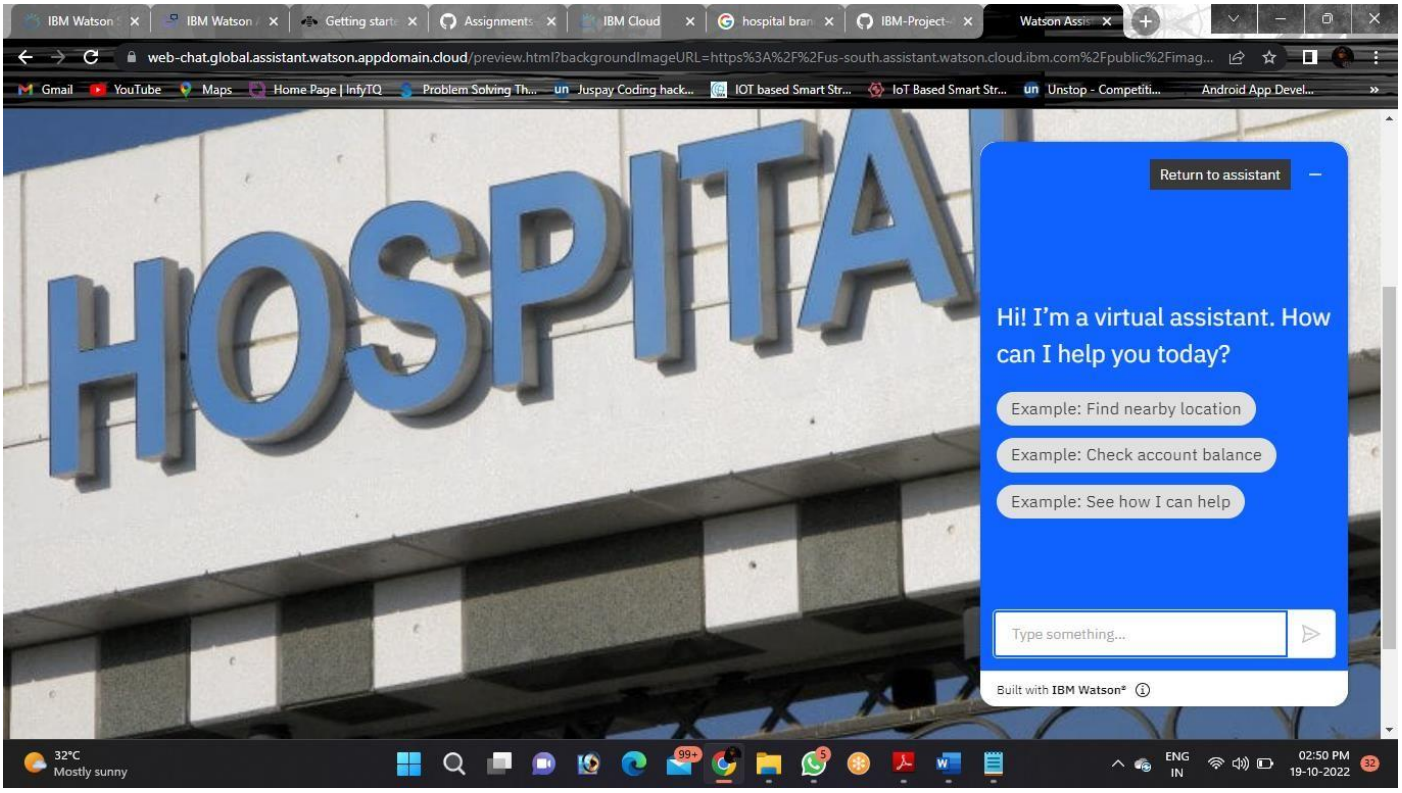

cloud.ibm.com/objectstorage/crn%3Av1%3Abluemix%3Apublic%3Acloud-object-storage%3Aglobal%3Aa%2F9ceab55aa9d8498abade4ea6d20f...
IBM Cloud
Search resources and products...
Catalog
Manage
Sowmiya L's Account
Cloud Object Storage
Storage instances
Cloud Object Storage-u6
Buckets
Integrations
Endpoints
Usage details
Service credentials
Connections
Plan
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
To delete/edit go to the IAM console.
Role for this bucket:
Content Reader
As a Content Reader, one can read and list objects in the bucket.
Create access policy
Type here to search
ENG
12:48 PM
18-10-2022

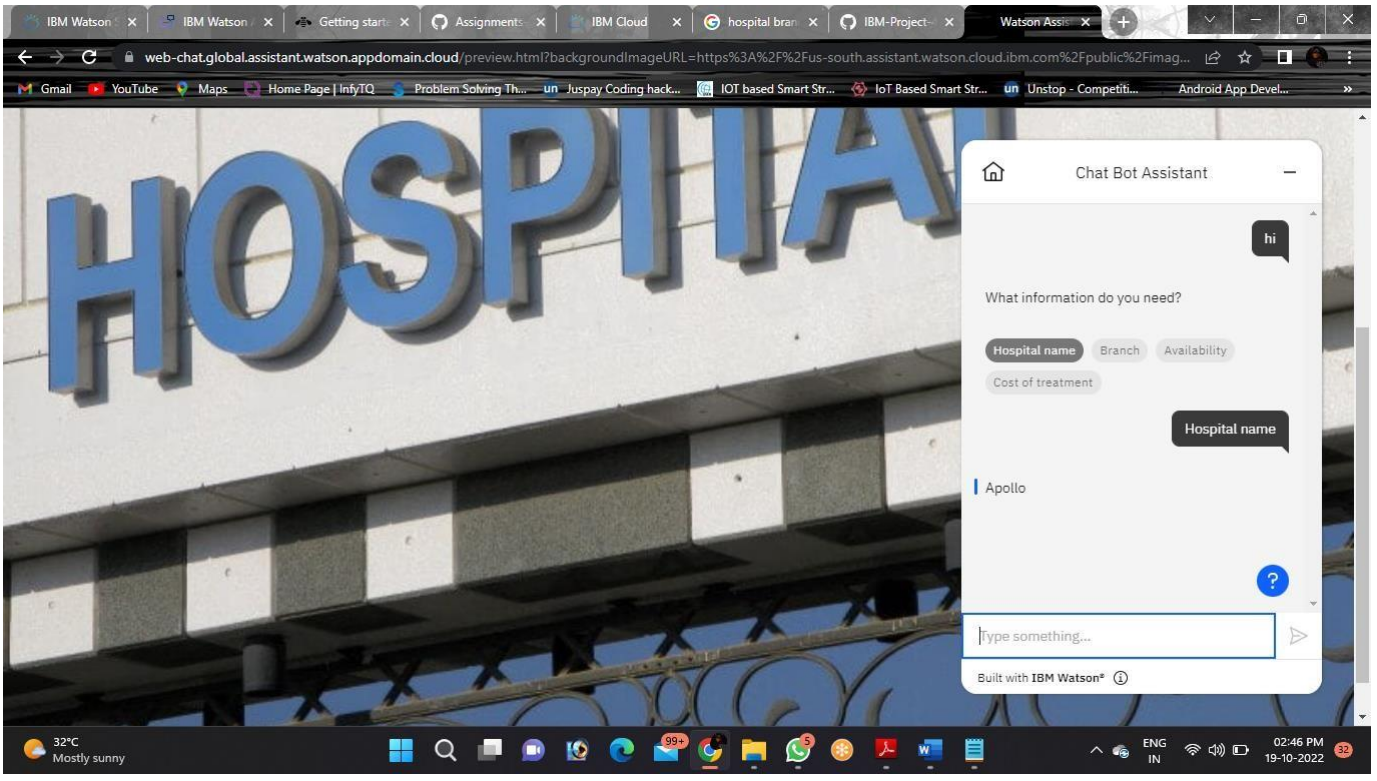
2. Upload a css page to the object storage and use the same page in your HTML code.





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-f7733c7b-3a1d-4617-b7cd-1bb799497d4c%3A%3Abb16aaec-7730-42ce-bc28-3d07d7f57d13&integrationID=c2c15a86-5873-4eca-9522-9f3494547f7b&region=us-south&serviceInstanceID=f7733c7b-3a1d-4617-b7cd-1bb799497d4c

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

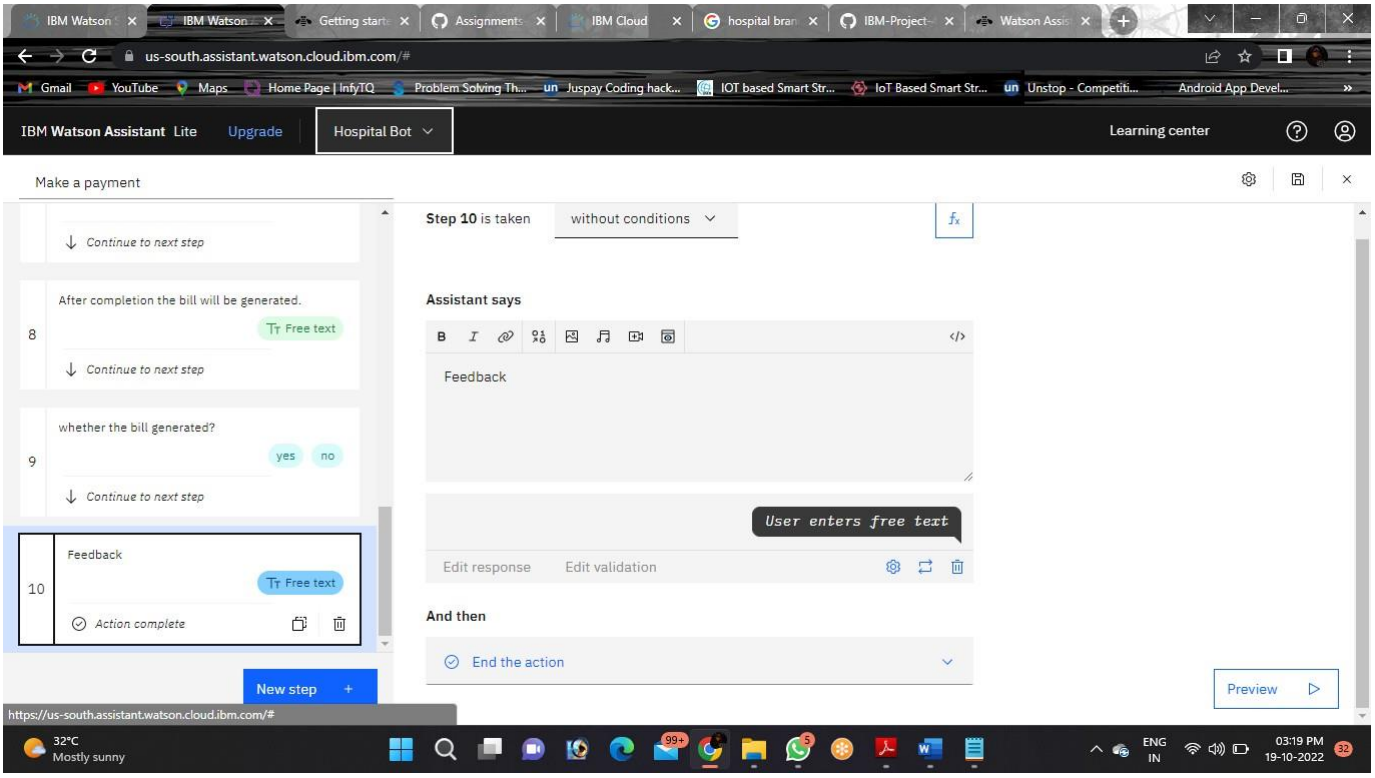
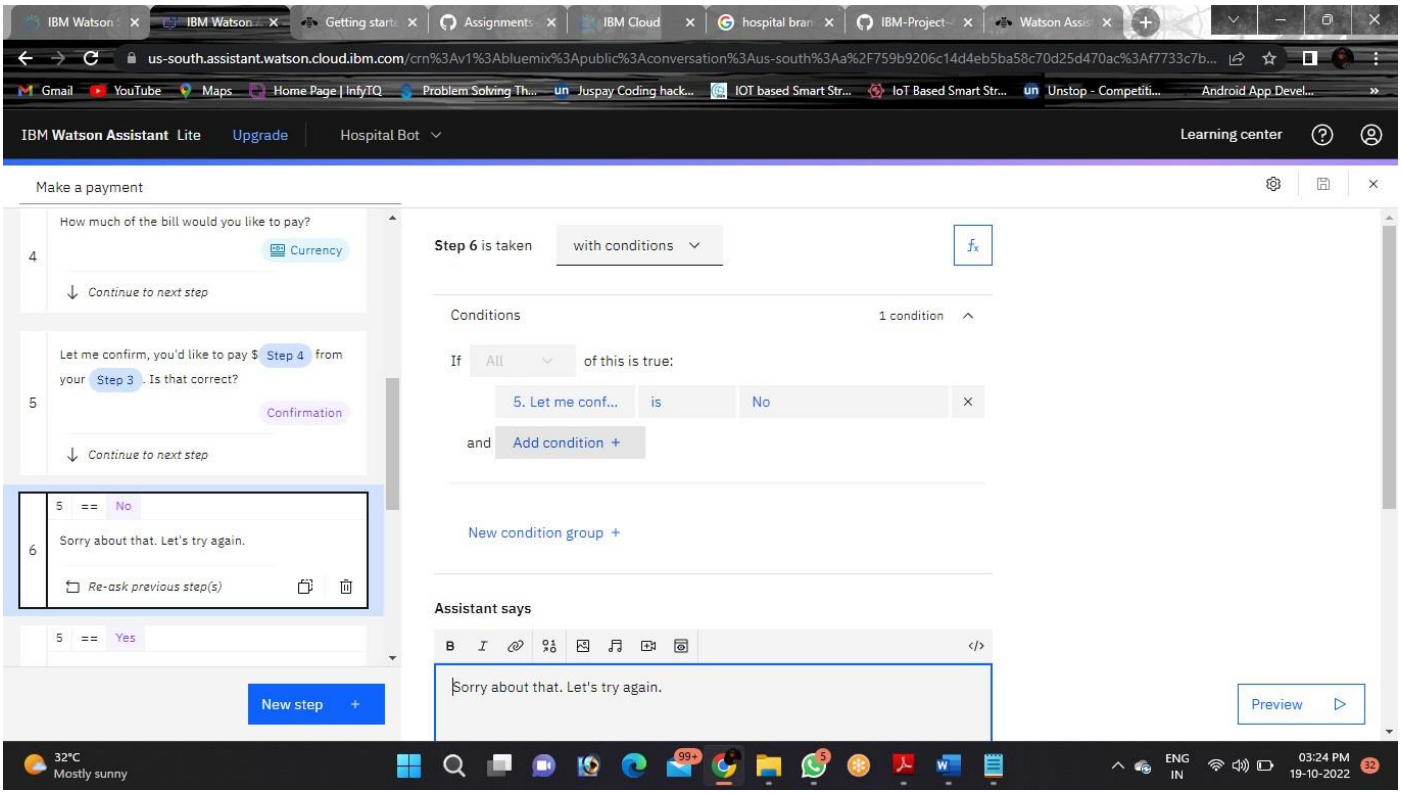
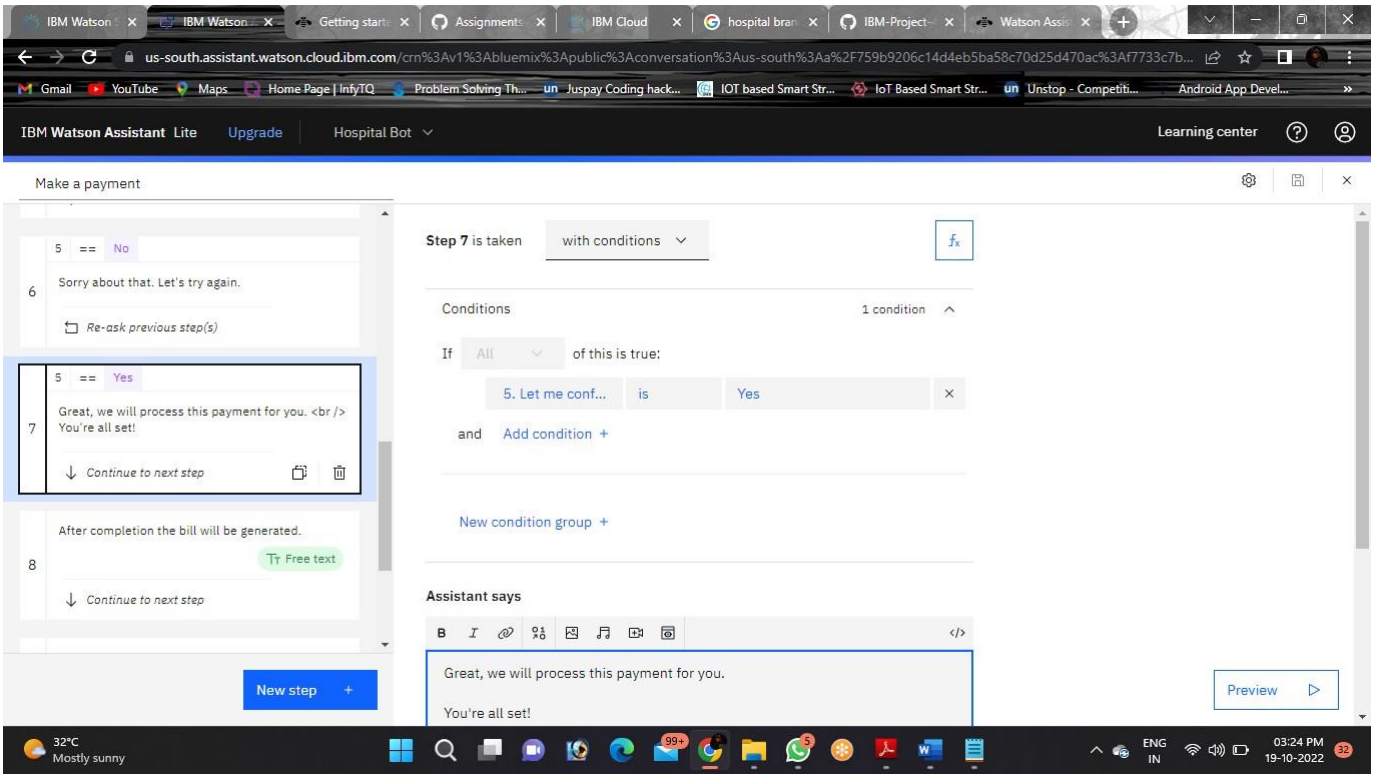
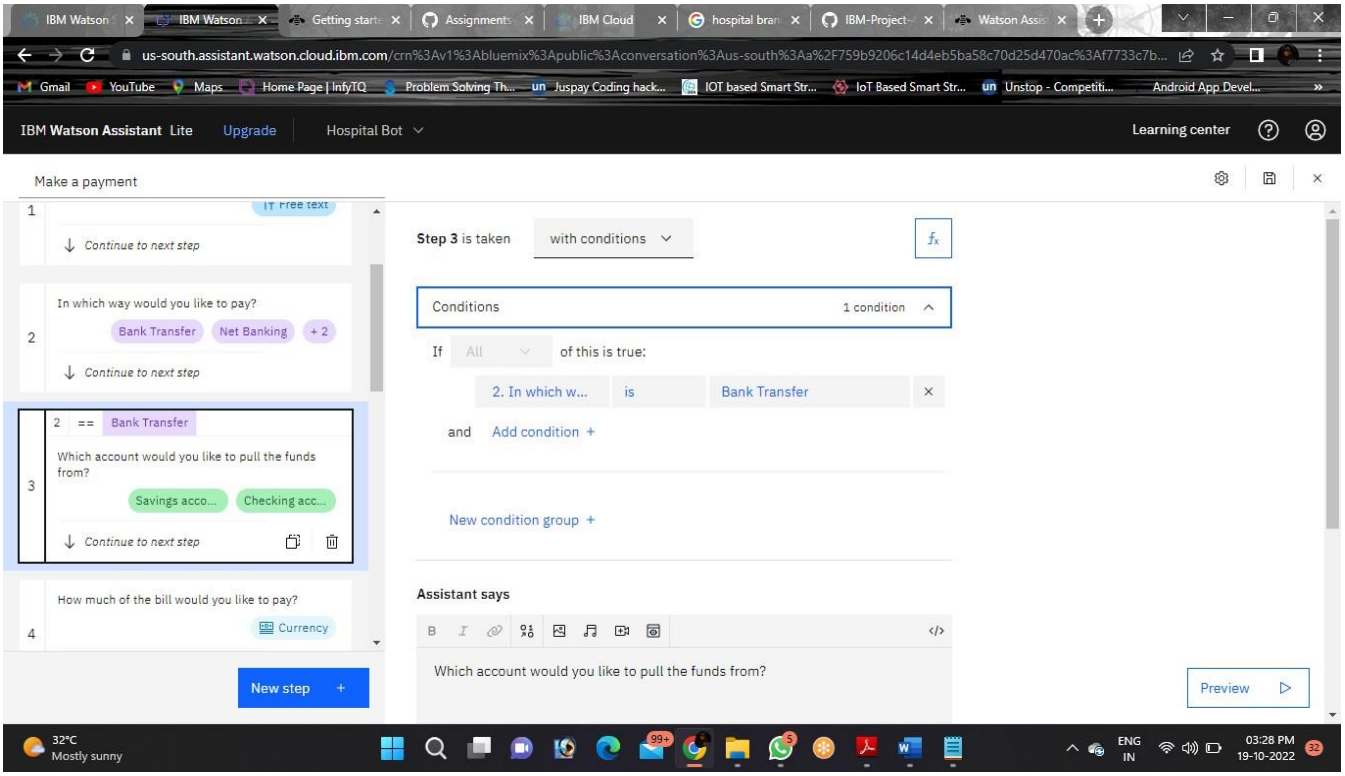


Figure 1. 10 steps of conversation

Included 3 conditions in steps:






IBM Watson Assistant Lite
Upgrade
Hospital Bot
Learning center
Make a payment
1
Continue to next step
In which way would you like to pay?
Bank Transfer
Net Banking
+ 2
2
Continue to next step
2 == Bank Transfer
Which account would you like to pull the funds from?
Savings acco...
Checking acc...
3
Continue to next step
How much of the bill would you like to pay?
Currency
4
New step
Step 3 is taken
with conditions
Conditions
1 condition
If All of this is true:
2. In which w...
is
Bank Transfer
and
Add condition +
New condition group +
Assistant says
Which account would you like to pull the funds from?
Preview
32°C
Mostly sunny
ENG
IN
03:28 PM
19-10-2022

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: "c2c15a86-5873-4eca-9522-9f3494547f7b", // The ID of this integration.
    region: "us-south", // The region your integration is hosted in.
    serviceInstanceID: "f7733c7b-3a1d-4617-b7cd-1bb799497d4c", // 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>
          <img src="https://cloudbucket.s3.jp-tok.cloud-object-storage.appdomain.cloud/{{row}}"
width="150px"></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
```