Assignment-3

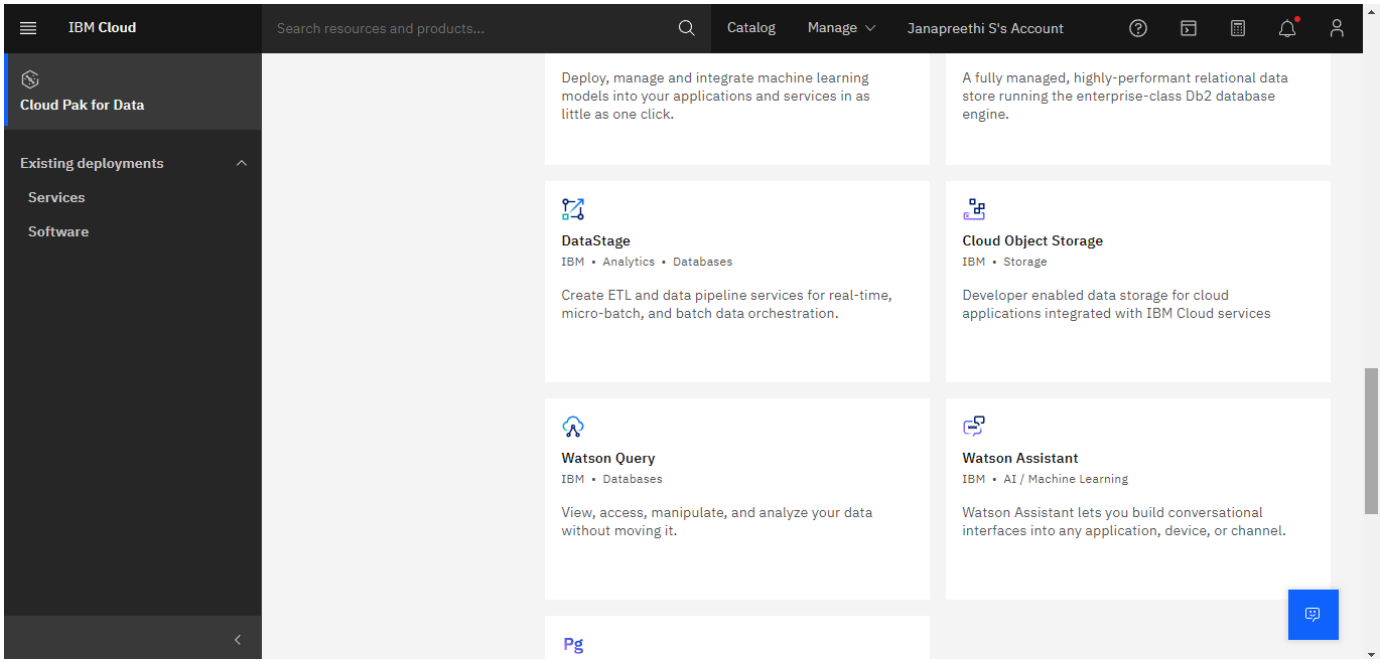
Name	Janapreethi S
Batch	B1-1M3E

1. Create a Bucket in IBM object storage.
2. Upload an 5 images to ibm object storage and make it public. write html code to displaying all the 5 images.
3. Upload a css page to the object storage and use the same page in your HTML code.
4. Design a chatbot using IBM Watson assistant for hospital. Ex: User comes with query to know the branches for that hospital in your city. Submit the web URL of that chat bot as a assignment.
5. Create Watson assistant service with 10 steps and use 3 conditions in it. Load that script in HTML page.

STEPS TO BE FOLLOWED

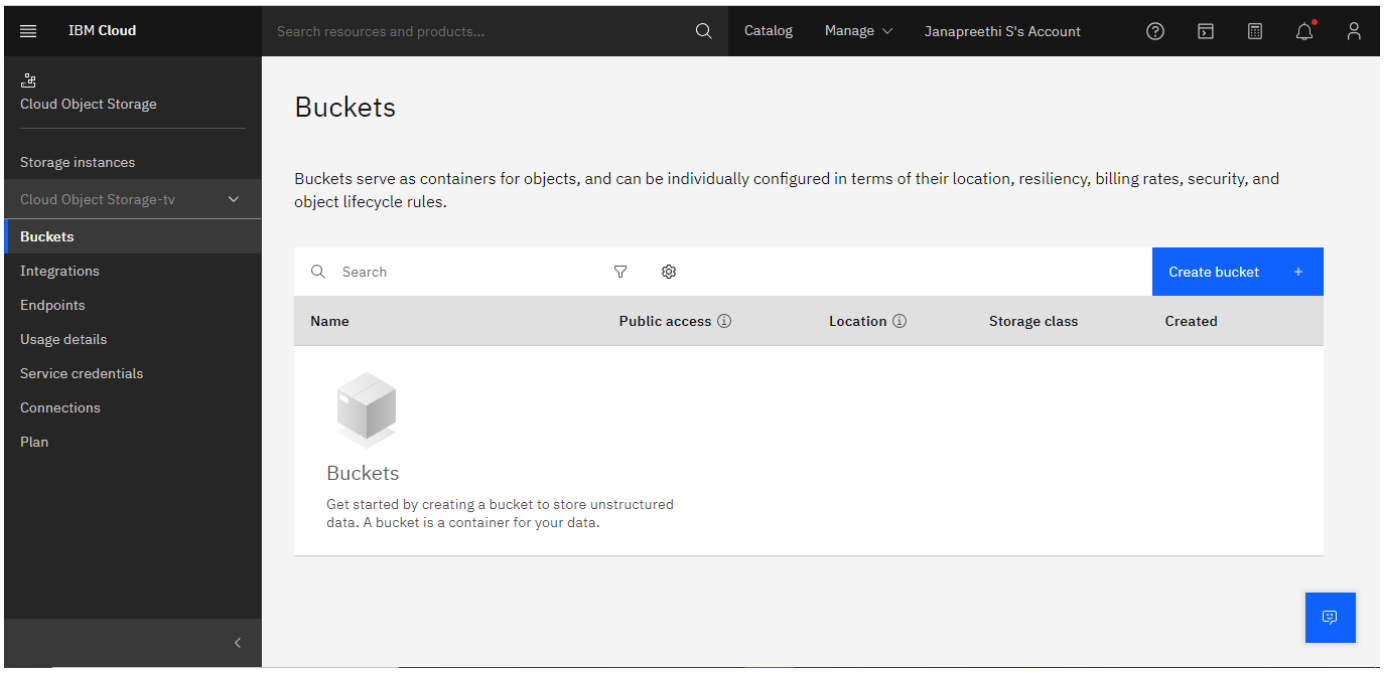
1. CREATE A BUCKET AND UPLOAD IMAGES

a) Select create resource button
b) Open *Object Storage* from the catalogue



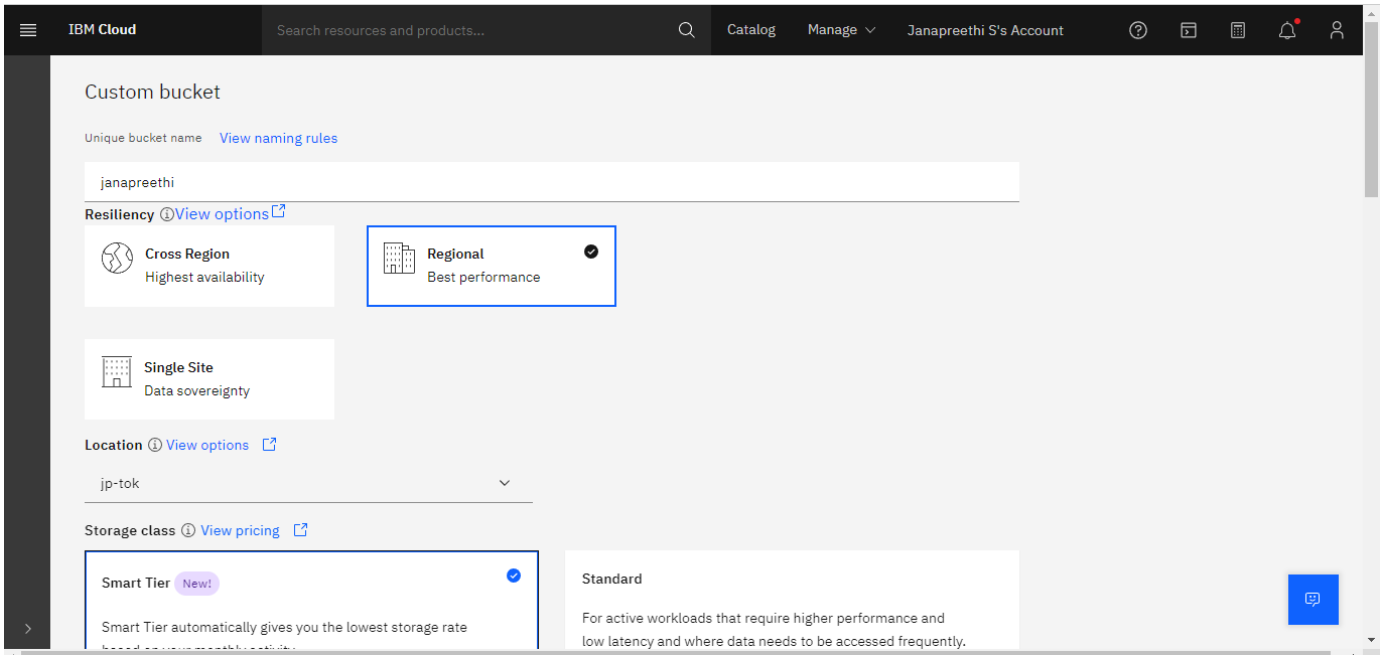
c) Choose *IBM Cloud* as infrastructure, select the free plan, give a name and select create
d) Go to active resource list and select the storage created

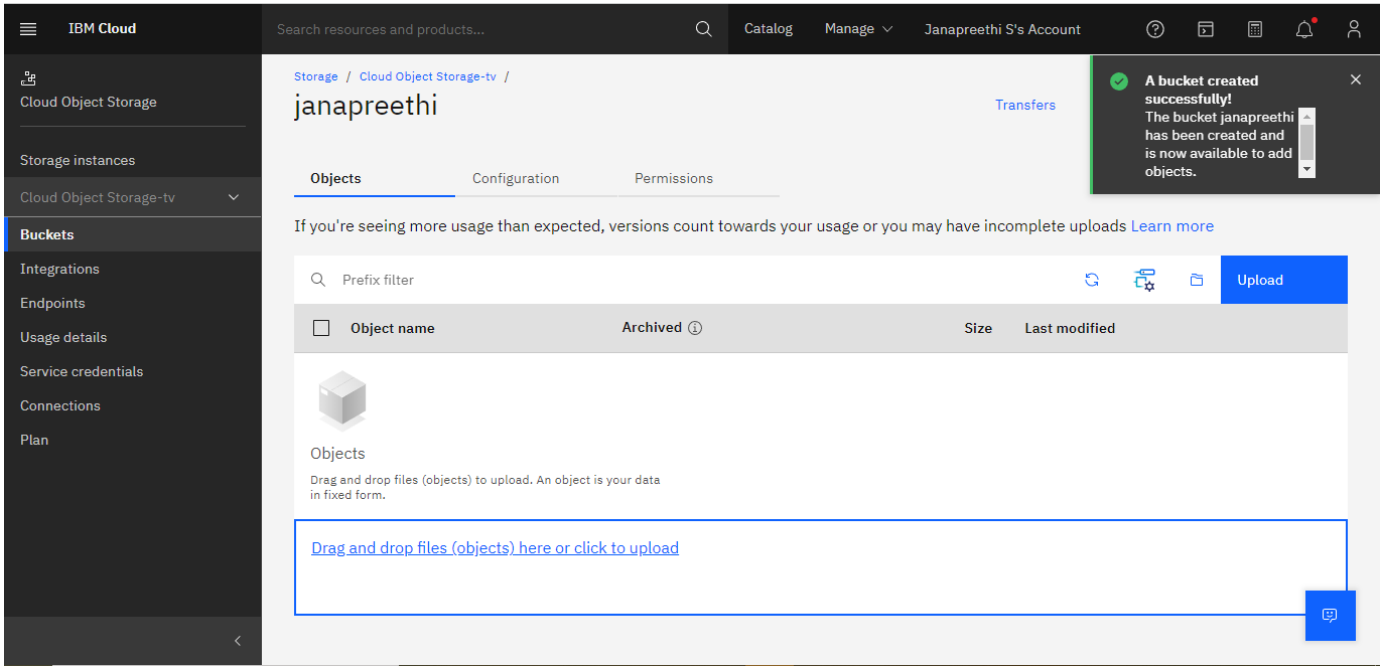
e) Select create bucket



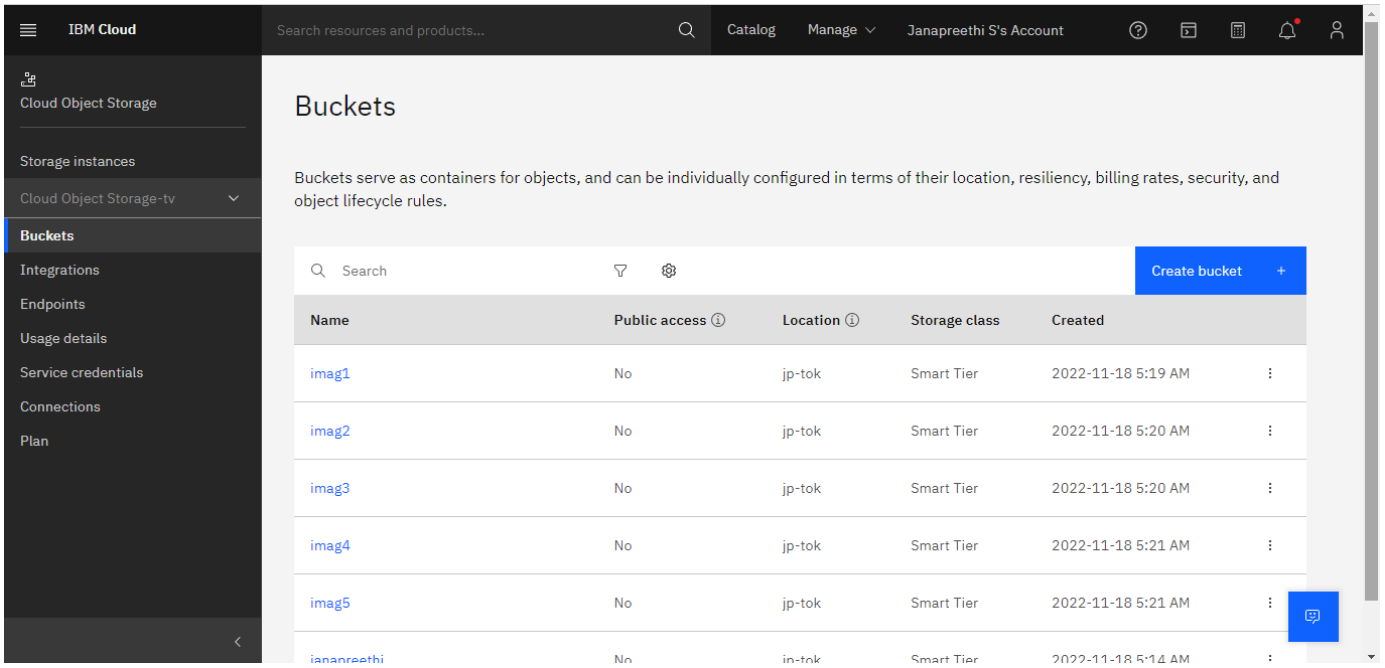
f) Select customize your bucket, give a unique name for the bucket across all available buckets around the globe

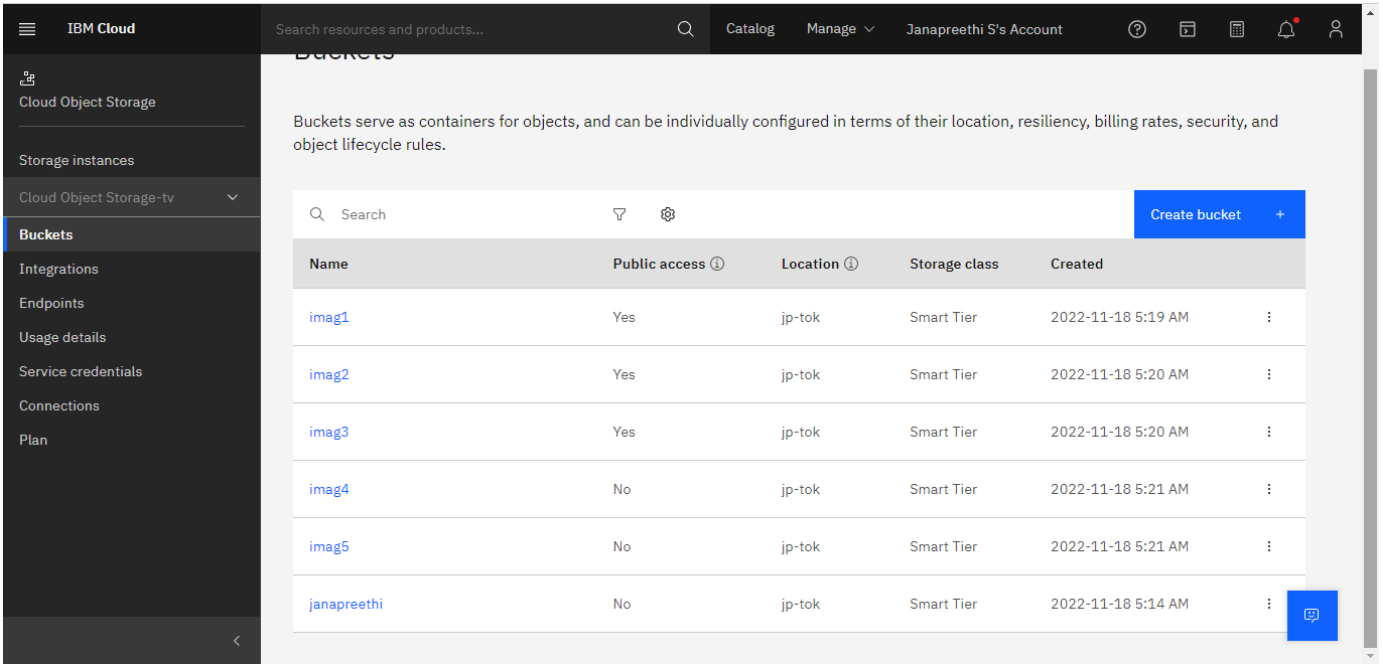
- Resiliency – Regional
- Storage class – Smart Tier

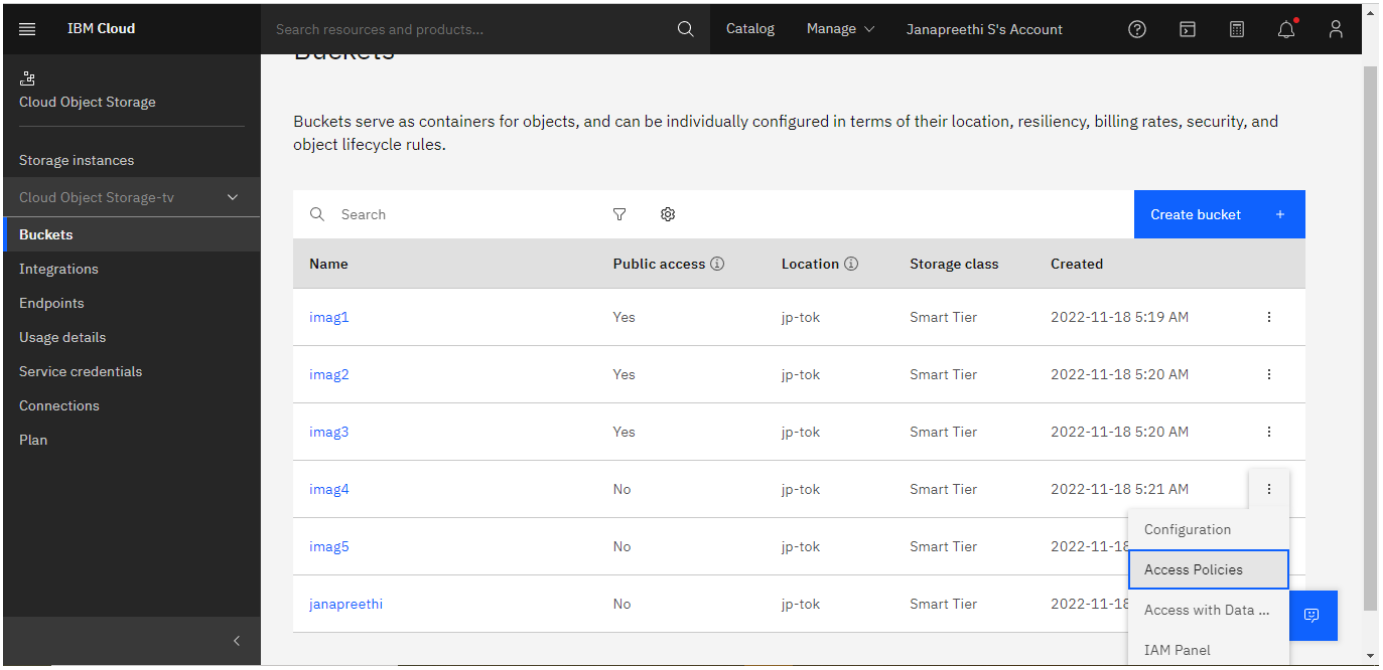


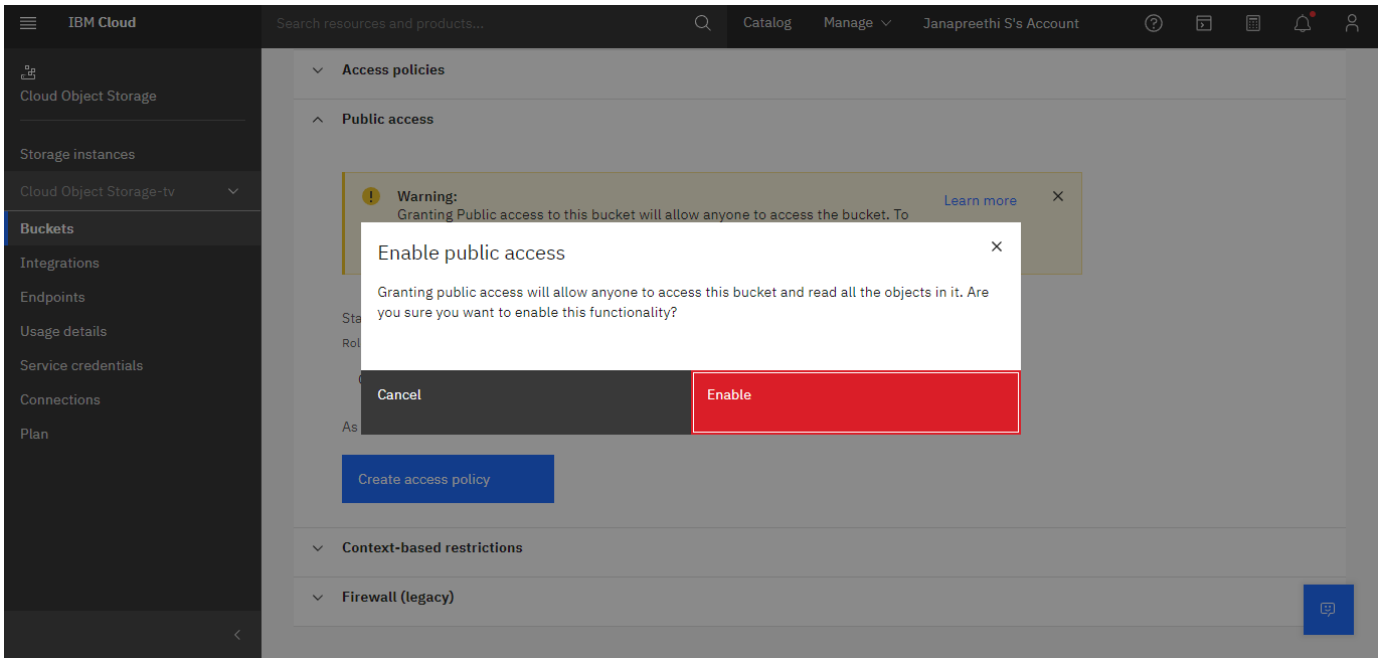


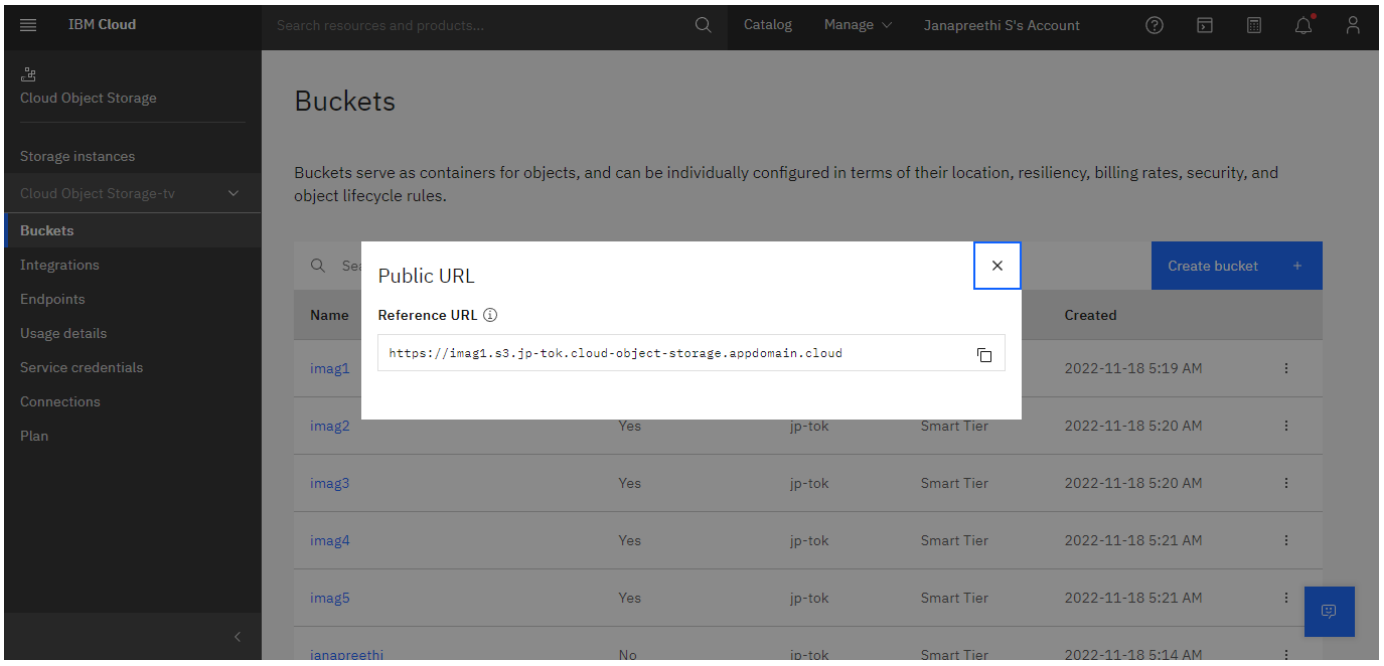
2. Upload an 5 images to ibm object storage and make it public. write html code to displaying all the 5 images.

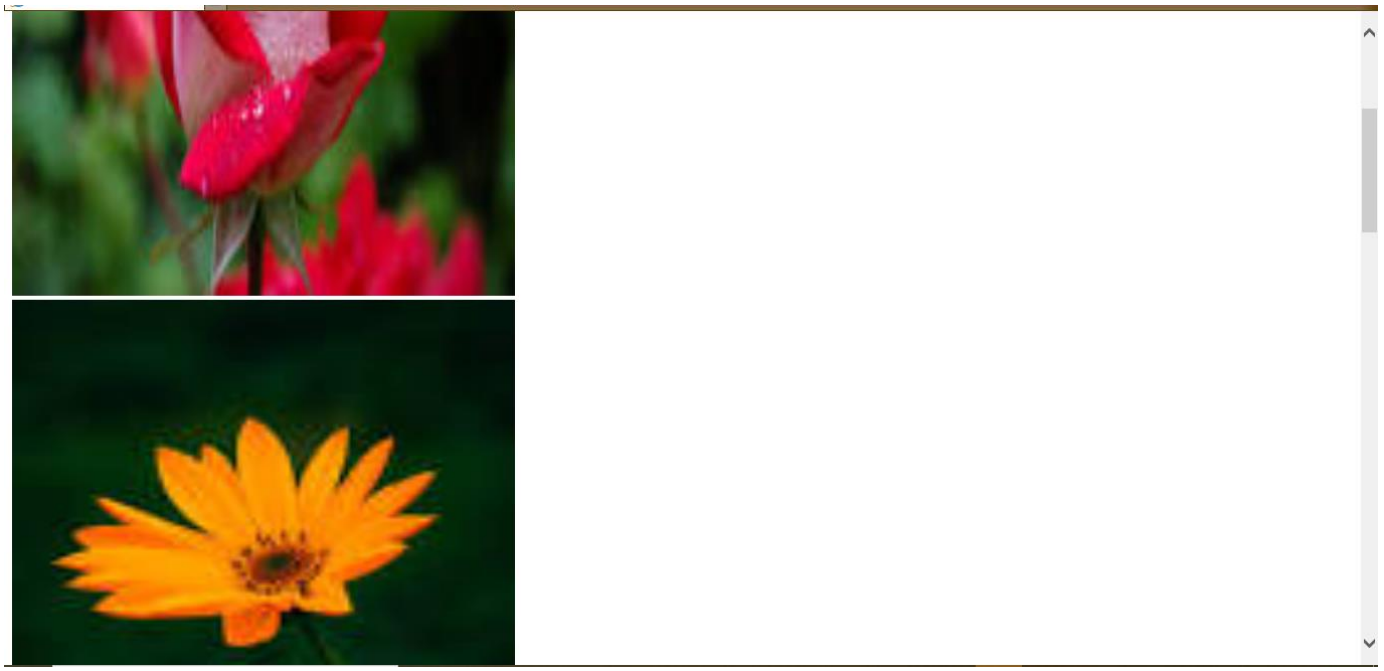



IBM Cloud
Search resources and products...
Catalog
Manage
Janapreethi S's Account
Cloud Object Storage
Storage instances
Cloud Object Storage-tv
Buckets
Integrations
Endpoints
Usage details
Service credentials
Connections
Plan
Buckets serve as containers for objects, and can be individually configured in terms of their location, resiliency, billing rates, security, and object lifecycle rules.
Search
Create bucket
Name
Public access
Location
Storage class
Created
imag1 Yes jp-tok Smart Tier 2022-11-18 5:19 AM
imag2 Yes jp-tok Smart Tier 2022-11-18 5:20 AM
imag3 Yes jp-tok Smart Tier 2022-11-18 5:20 AM
imag4 No jp-tok Smart Tier 2022-11-18 5:21 AM
imag5 No jp-tok Smart Tier 2022-11-18 5:21 AM
janapreethi No jp-tok Smart Tier 2022-11-18 5:14 AM

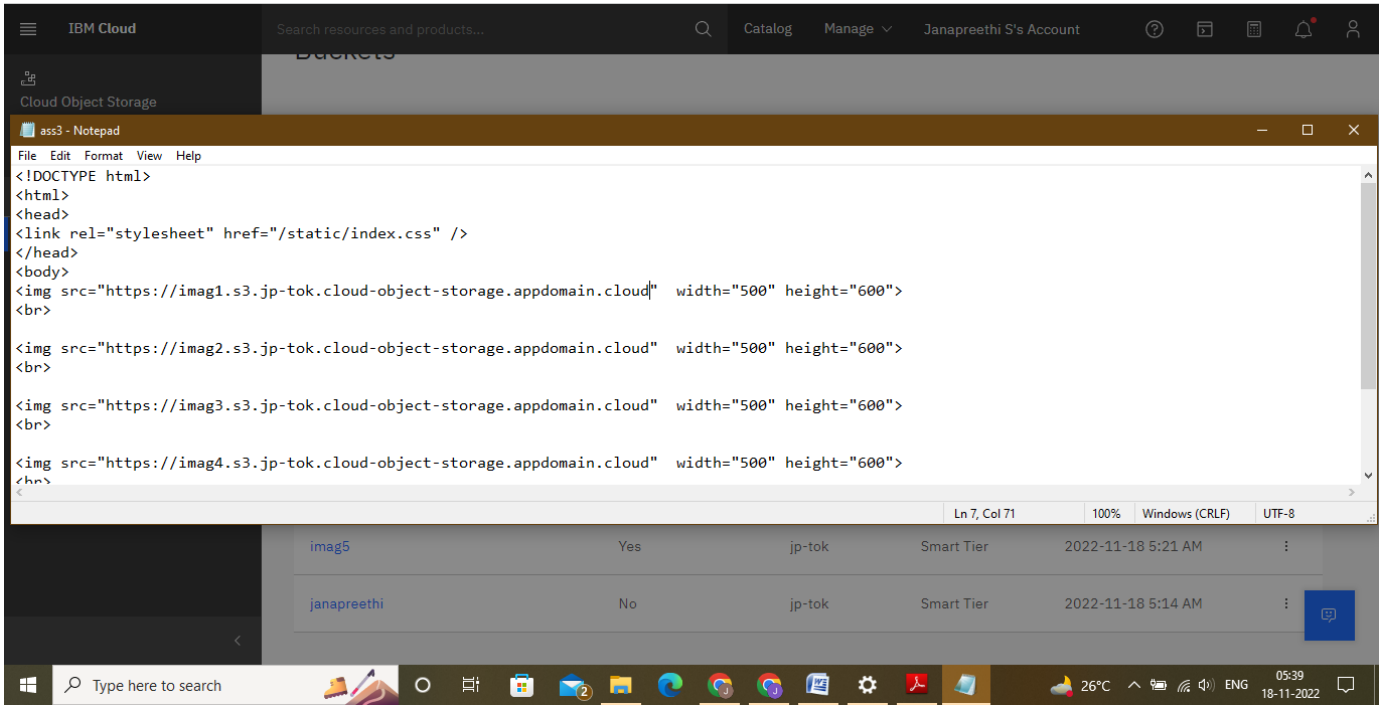

IBM Cloud
Search resources and products...
Catalog
Manage
Janapreethi S's Account
Cloud Object Storage
Storage instances
Cloud Object Storage-tv
Buckets
Integrations
Endpoints
Usage details
Service credentials
Connections
Plan
Buckets serve as containers for objects, and can be individually configured in terms of their location, resiliency, billing rates, security, and object lifecycle rules.
Search
Create bucket
Name
Public access
Location
Storage class
Created
imag1 Yes jp-tok Smart Tier 2022-11-18 5:19 AM
imag2 Yes jp-tok Smart Tier 2022-11-18 5:20 AM
imag3 Yes jp-tok Smart Tier 2022-11-18 5:20 AM
imag4 No jp-tok Smart Tier 2022-11-18 5:21 AM
imag5 No jp-tok Smart Tier
janapreethi No jp-tok Smart Tier
Configuration
Access Policies
Access with Data ...
IAM Panel

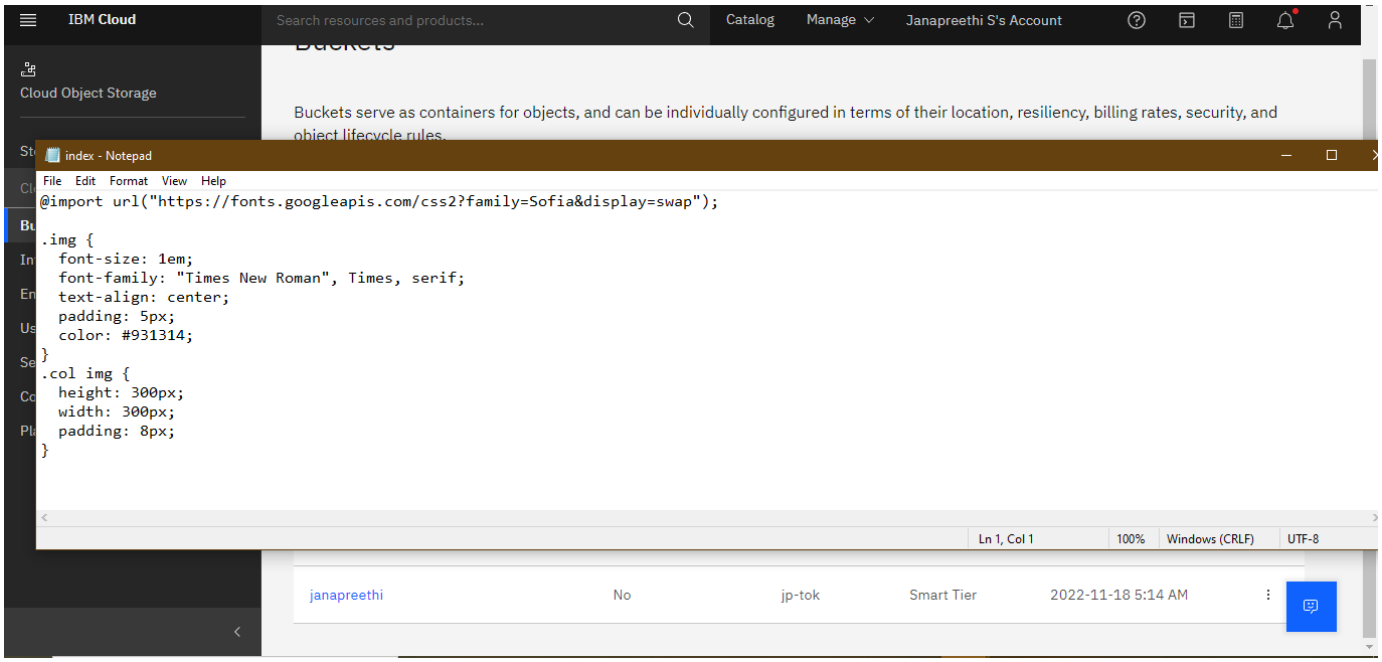

IBM Cloud
Search resources and products...
Catalog
Manage
Janapreethi S's Account
Cloud Object Storage
Storage instances
Cloud Object Storage-tv
Buckets
Integrations
Endpoints
Usage details
Service credentials
Connections
Plan
Access policies
Public access
Warning:
Granting Public access to this bucket will allow anyone to access the bucket. To
Learn more
Enable public access
Granting public access will allow anyone to access this bucket and read all the objects in it. Are you sure you want to enable this functionality?
Cancel
Enable
Create access policy
Context-based restrictions
Firewall (legacy)


IBM Cloud
Search resources and products...
Catalog
Manage
Janapreethi S's Account
Cloud Object Storage
Storage instances
Cloud Object Storage-tv
Buckets
Integrations
Endpoints
Usage details
Service credentials
Connections
Plan
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.
Public URL
Reference URL
https://imag1.s3.jp-tok.cloud-object-storage.appdomain.cloud
Create bucket
Name
Created
imag1
2022-11-18 5:19 AM
imag2
Yes
jp-tok
Smart Tier
2022-11-18 5:20 AM
imag3
Yes
jp-tok
Smart Tier
2022-11-18 5:20 AM
imag4
Yes
jp-tok
Smart Tier
2022-11-18 5:21 AM
imag5
Yes
jp-tok
Smart Tier
2022-11-18 5:21 AM
janapreethi
No
jp-tok
Smart Tier
2022-11-18 5:14 AM



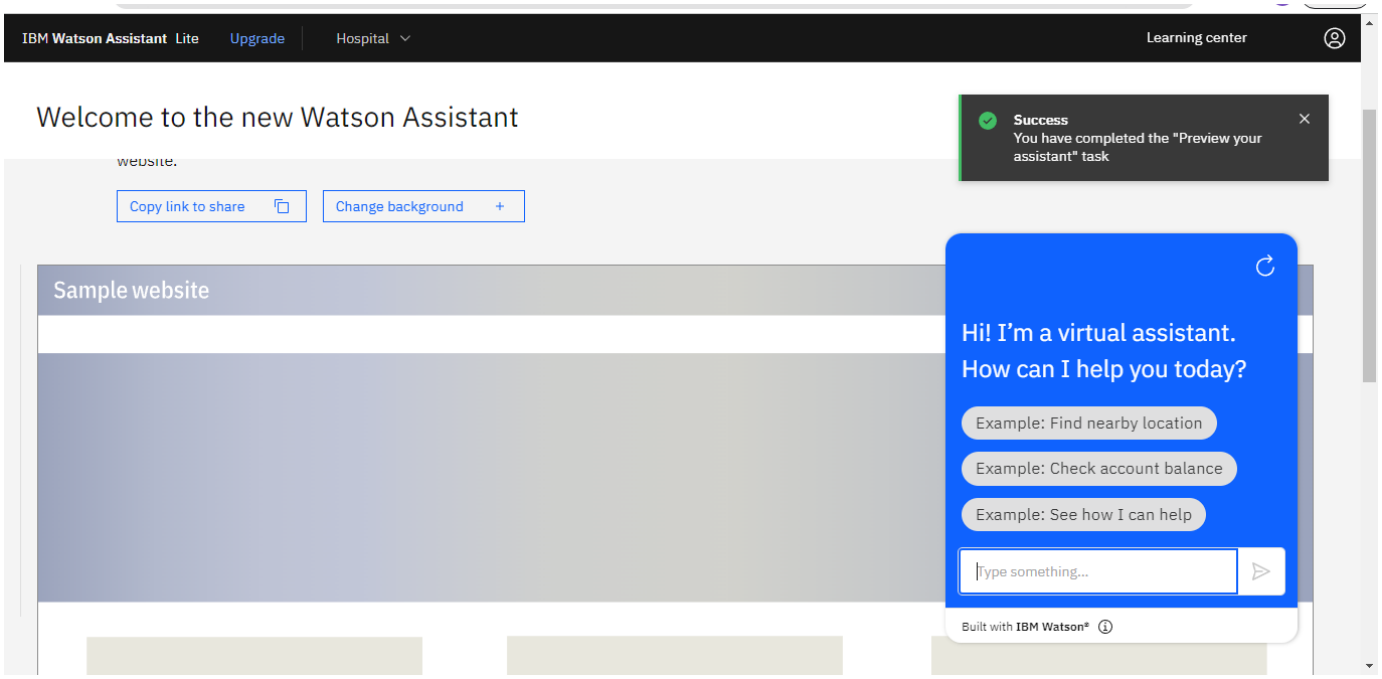
3. Upload a css page to the object storage and use the same page in your HTML code.

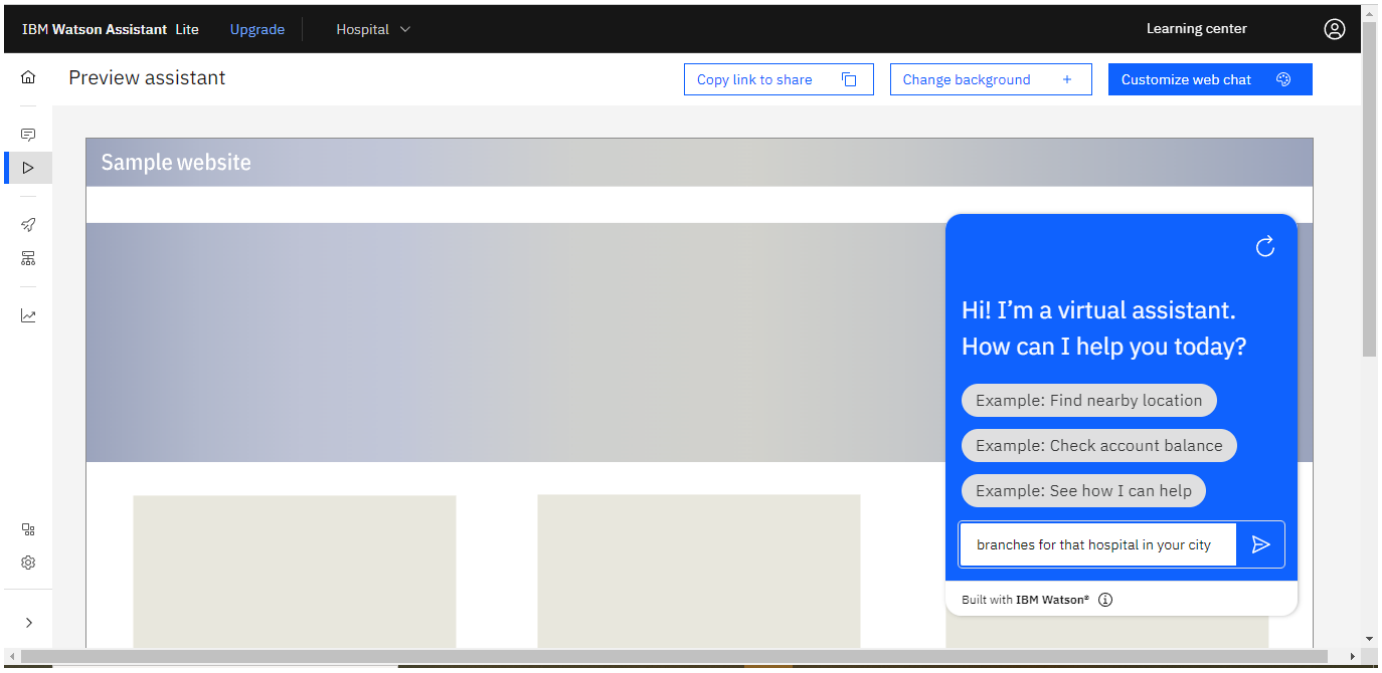


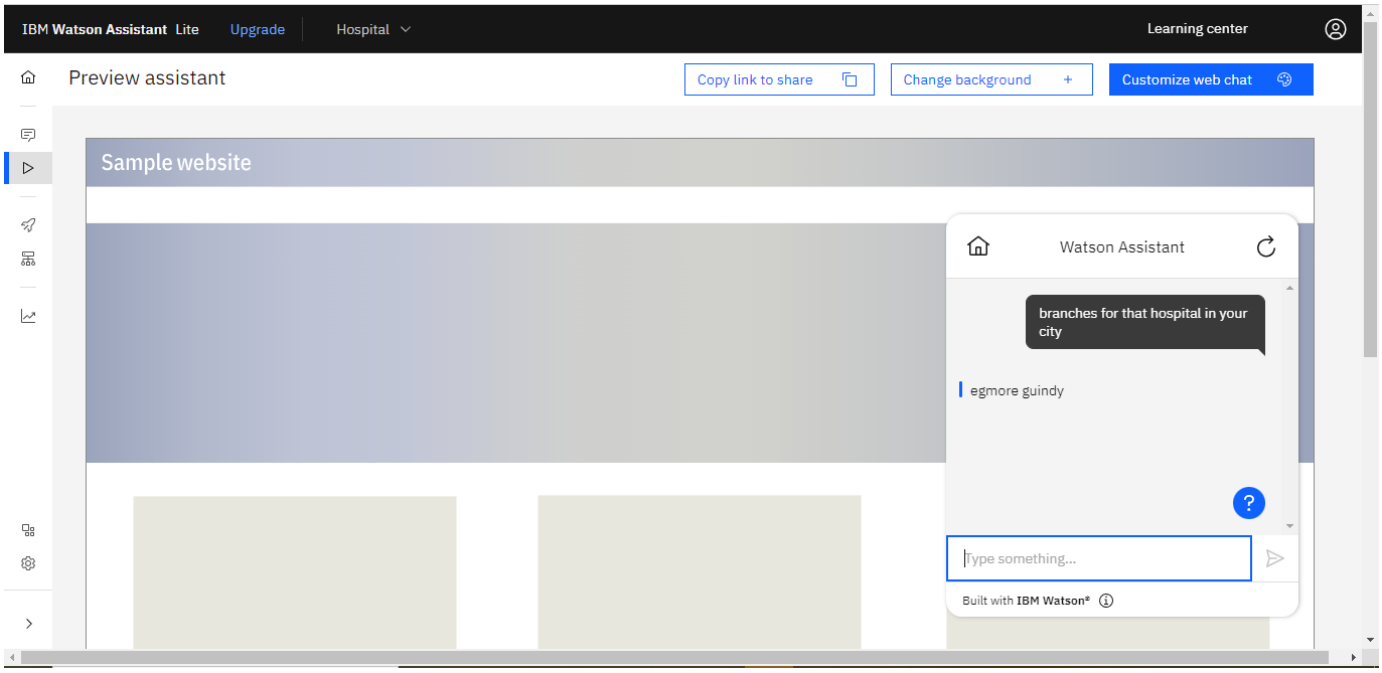


4. Design a chatbot using IBM Watson assistant for hospital

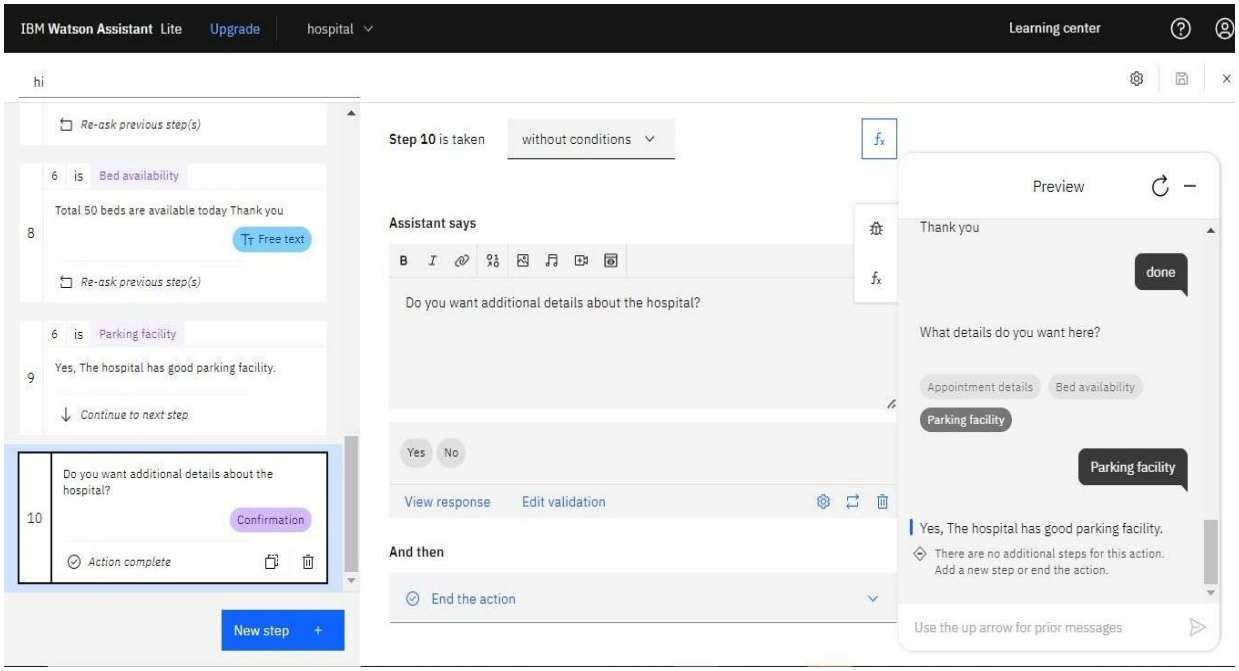
a) Select create resource button
b) Open *Watson Assistant* from the catalogue
c) Choose the lite plan and create the bot
d) Launch the assistant and select the actions tab to add new actions to the bot



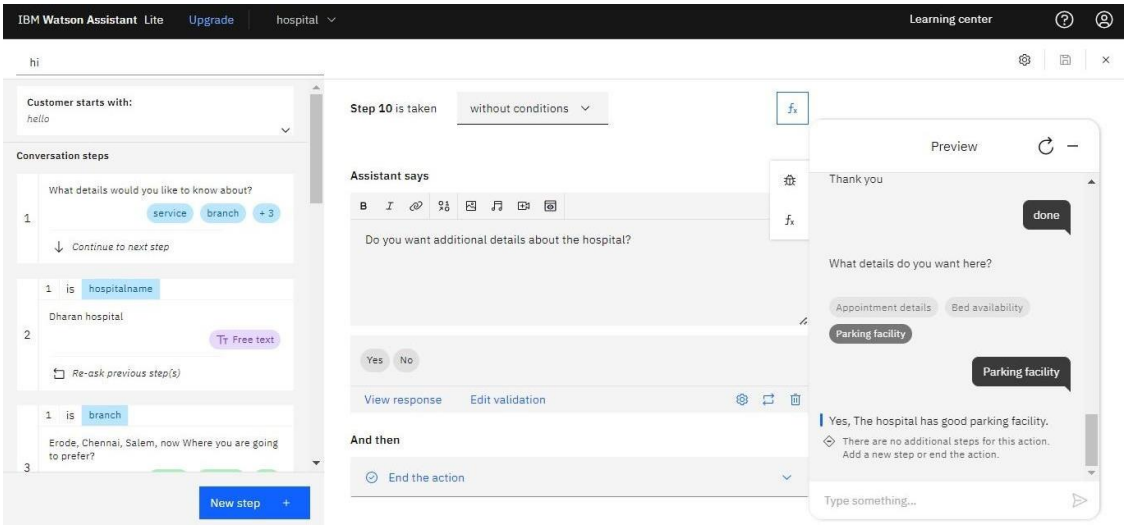


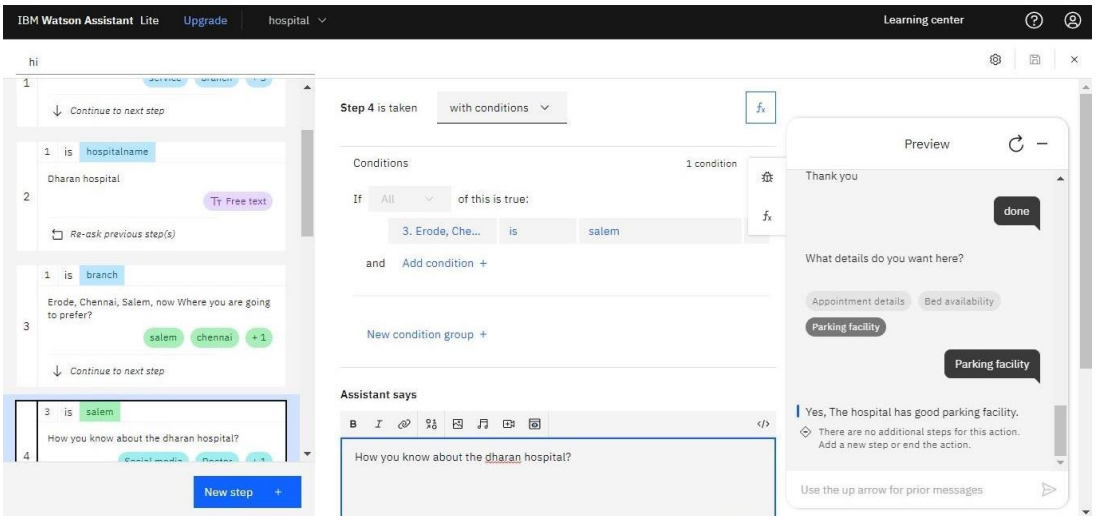


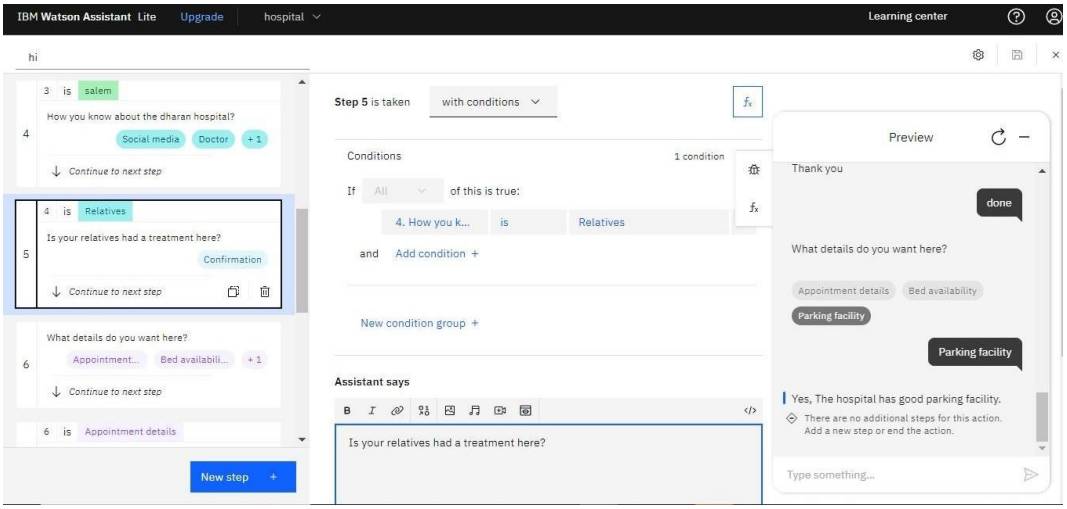
5. Create Watson assistant service with 10 steps and use 3 conditions in it. Load that script in HTML page.

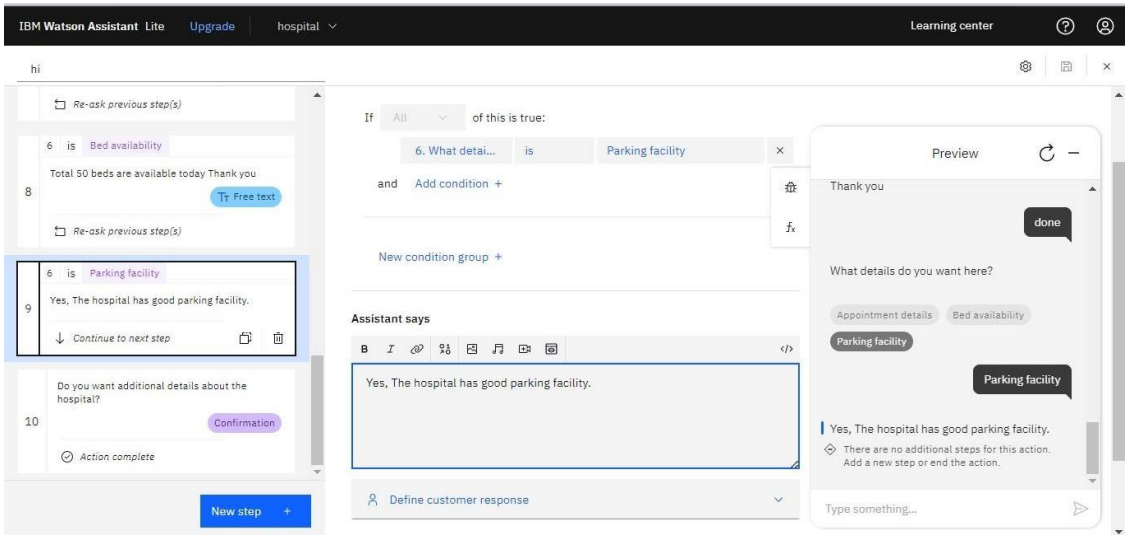


Included 3 conditions in steps:









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: "8c0d878c-d3b4-4a3e-b63a-01b255dc7d6e", // The ID of this integration.region: "us-
     south", // The region your integration is hosted in.
     serviceInstanceID:       "398c4efa-af81-42f3-badd-b3b31e98f373",
     // 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>
  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,requestimport
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)
```