

## Assignment2

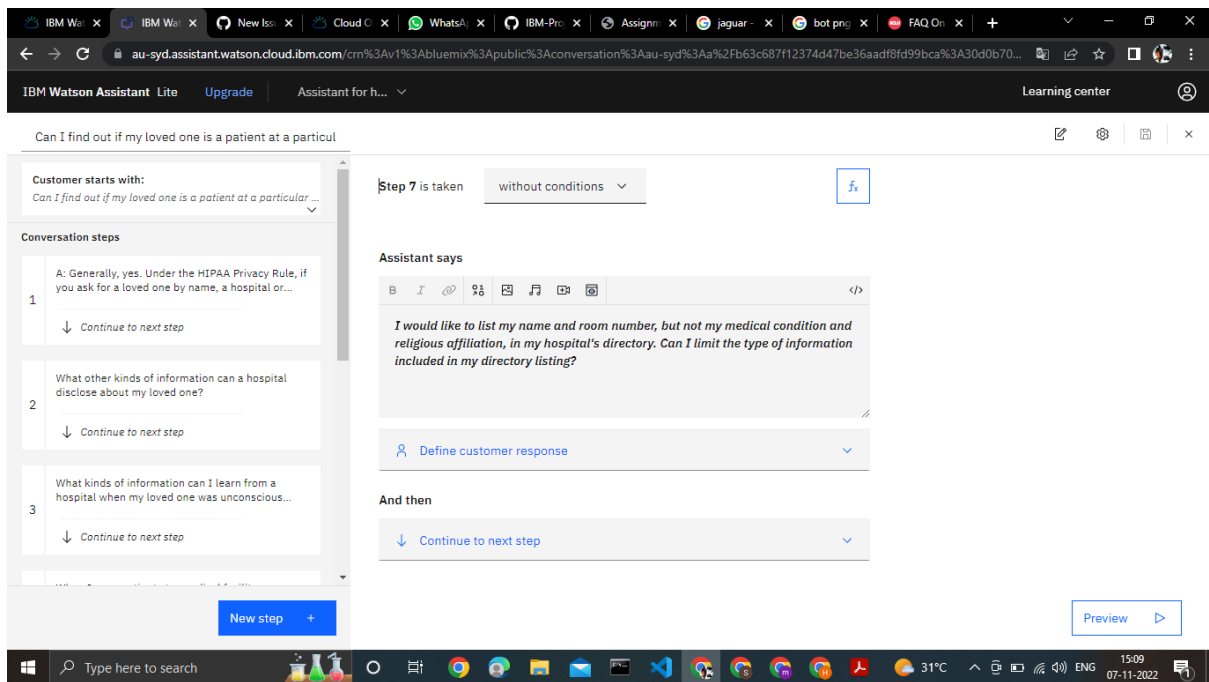
### Project Title: Inventory Management System for Retailers

Team ID: PNT2022TMID43190

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

The screenshot shows the IBM Watson Assistant interface. The left sidebar displays a list of steps in a conversation flow. Step 7 is currently selected. The main area shows the details for Step 7, which is titled "Step 7 is taken" and has a dropdown menu set to "without conditions". The "Assistant says" section contains a text box with the following text: "I would like to list my name and room number, but not my medical condition and religious affiliation, in my hospital's directory. Can I limit the type of information included in my directory listing?". Below this, there is a "Define customer response" button. The "And then" section has a dropdown menu set to "Continue to next step". The bottom of the interface shows a Windows taskbar with various application icons and the system clock indicating 15:09 on 07-11-2022.

The screenshot shows the IBM Watson Assistant interface. The left sidebar displays a list of steps in a conversation flow. Step 5 is currently selected. The main area shows the details for Step 5, which is titled "Step 5 is taken" and has a dropdown menu set to "without conditions". The "Assistant says" section contains a text box with the following text: "I do not want to share my information with anyone - not even my closest family members. Can I request to have my information excluded from my hospital's directory?". Below this, there is a "Define customer response" button. The "And then" section has a dropdown menu set to "Continue to next step". The bottom of the interface shows a Windows taskbar with various application icons and the system clock indicating 15:09 on 07-11-2022.



## 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: "14b83b8f-3dfd-405f-9520-
b550092892aa", // The ID of this integration. region: "us-south", // The region your
integration is hosted in. serviceInstanceID: "6e95bee9-8d0b-49f6-8a2f-4125fb3a7945", //
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(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=['P
OST']) def upload():

    name_file=request.for
    m['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
```