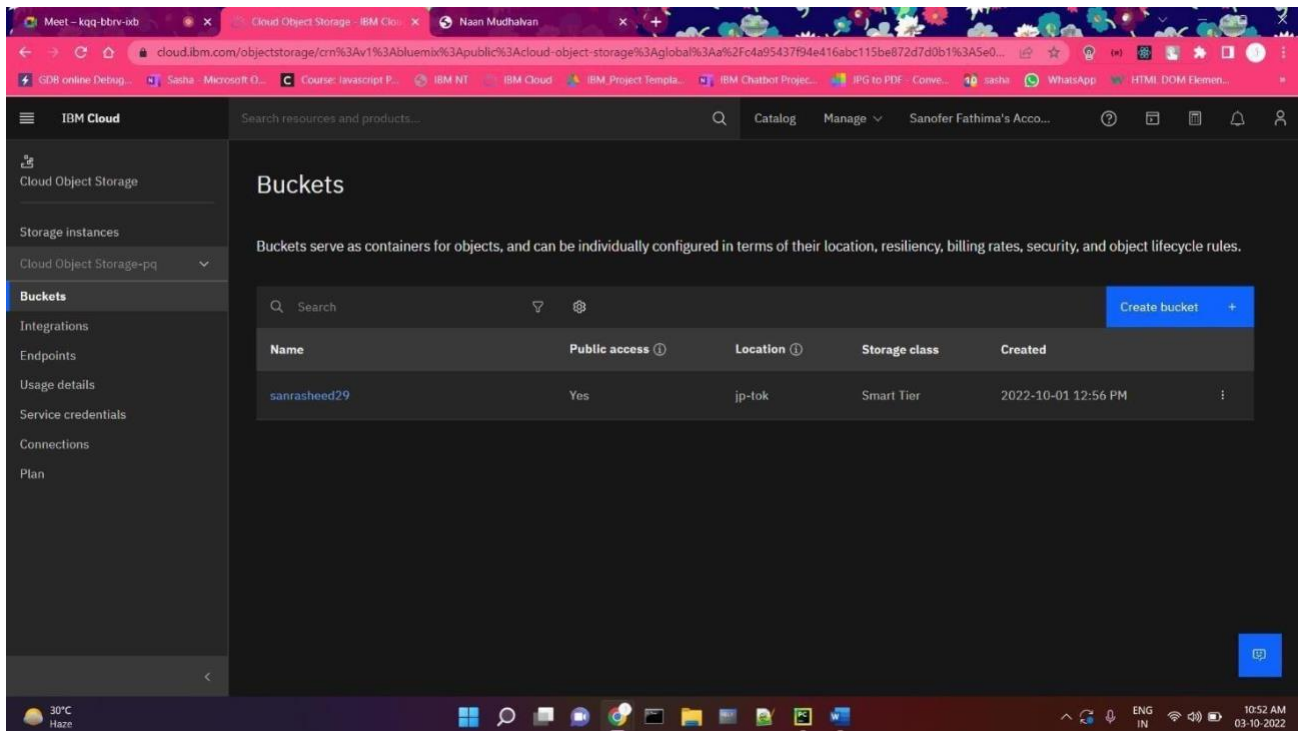


ASSIGNMENT 3

CLOUD APPLICATION DEVELOPMENT

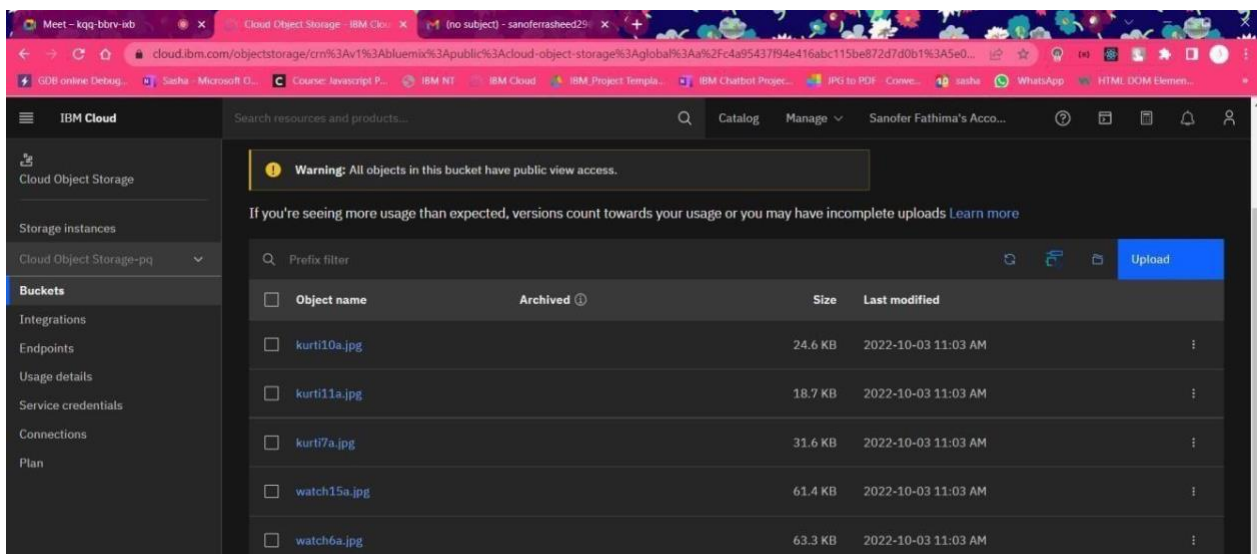
Student Name	V.Natesh krishnan
Student Roll Number	91272019104102
Maximum Marks	2 Marks

1. Create a Bucket in IBM object storage.



Thus a bucket is created with an unique name sanrasheed29.

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



Thus, Five images are uploaded and access is made public.

PYTHON CODE:

```
from flask import Flask,render_template,request,redirect,url_for
from connect import *
from objstorage import *
app= Flask(_name )
```

```
@app.route("/assign3")
def objstorage():
    for i in range(0,len(imagearr)):
        imagearr[i]="https://s3.jp-tok.cloud-object-
storage.appdomain.cloud/sanrasheed29/"+imagearr[i]
    print("objstorage ", imagearr)
    return render_template('assign3.html',arr=imagearr)
```

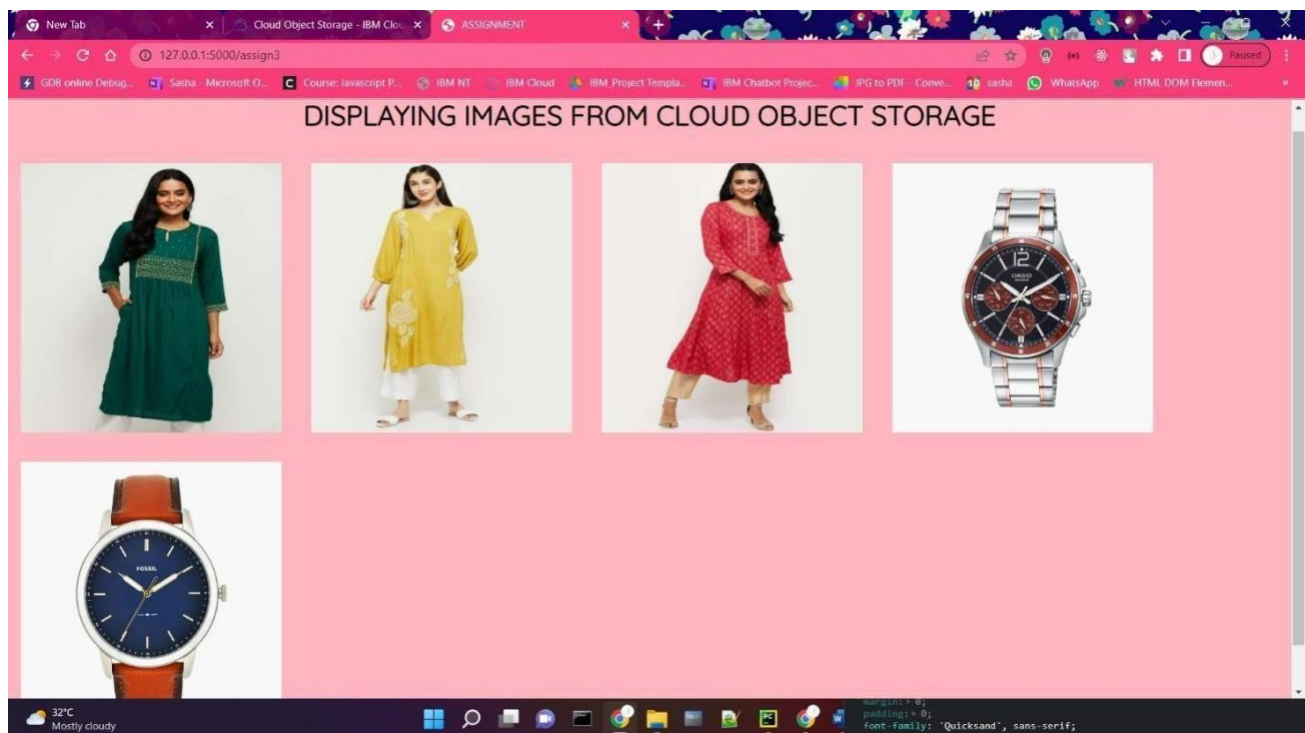
```
@app.route('/redirect_to')
def redirect_to():
    link = request.args.get('link', '/')
    return redirect(link), 301
```

```
if(_name_=='_main_'):
    app.run()
```

HTML CODE TO DISPLAY IMAGES:

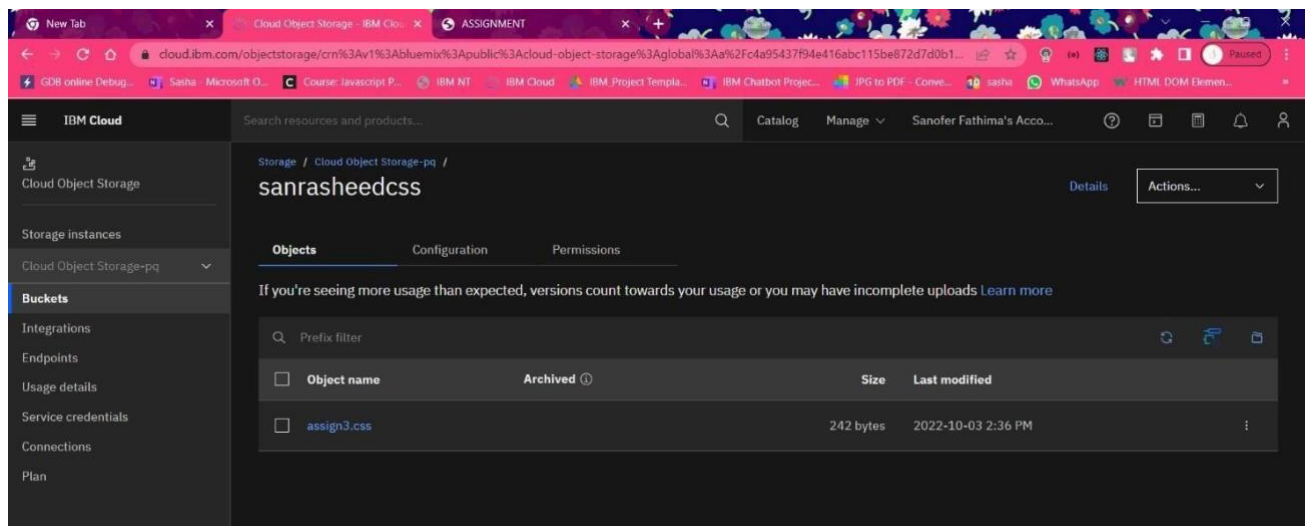
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>ASSIGNMENT</title>
    <link rel="stylesheet" href="{{url_for('redirect_to',link='https://s3.jp-tok.cloud-object-
storage.appdomain.cloud/sanrasheedcss/assign3.css')}}" type="text/css">
</head>
<body >
<h1>DISPLAYING IMAGES FROM CLOUD OBJECT STORAGE</h1>
{% for image in arr: %}

{% endfor %}
</body>
</html>
```

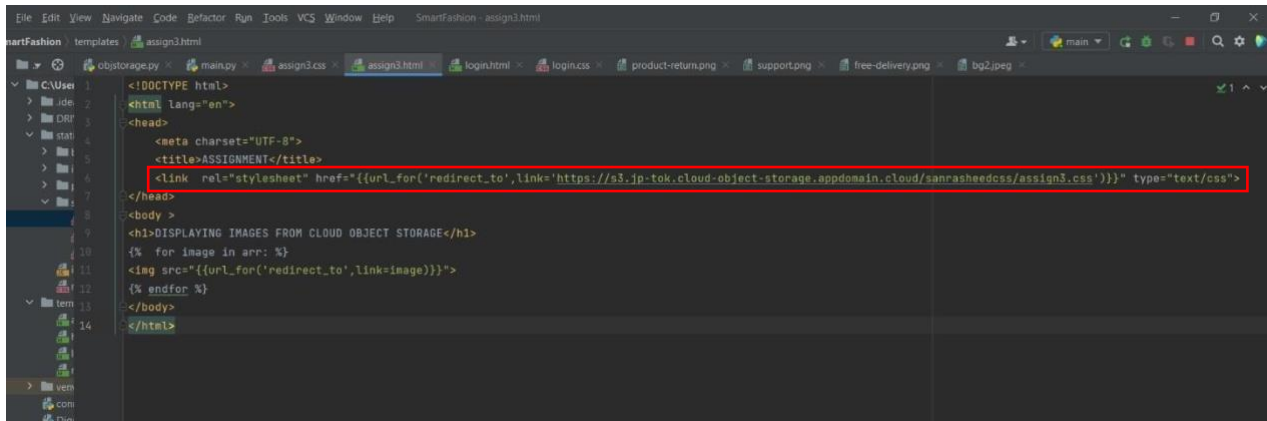


Thus the 5 images stored in cloud object storage has been displayed in the website.

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



A CSS page is uploaded in object storage.

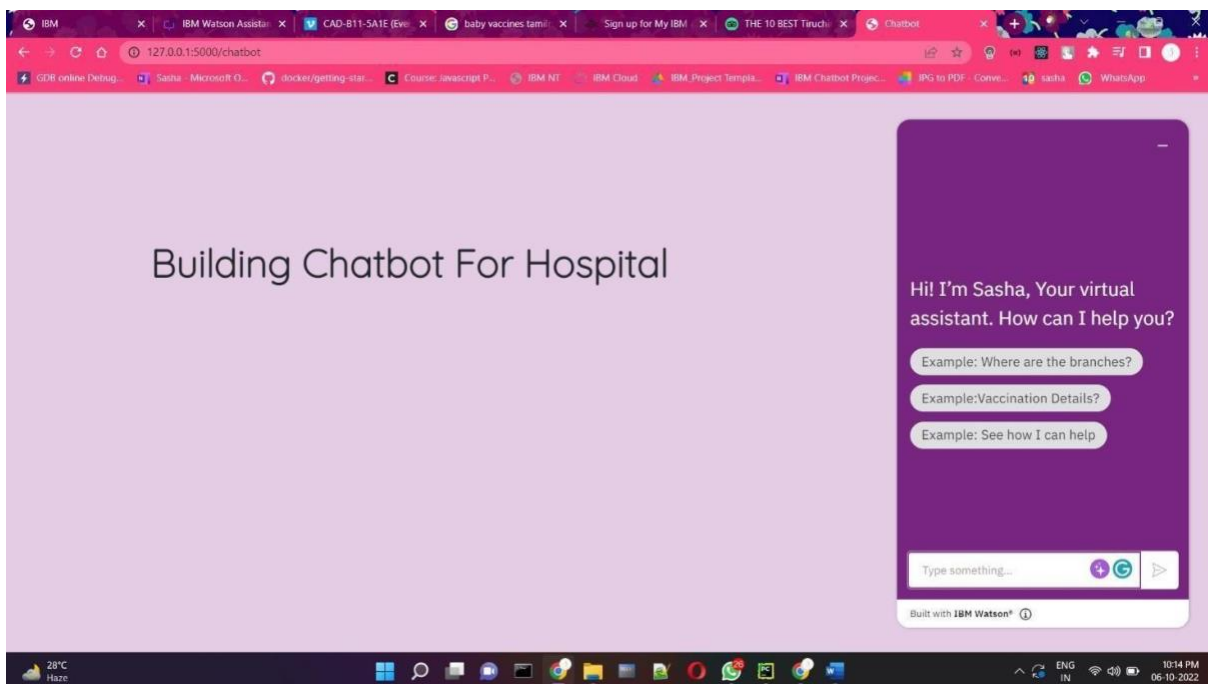


```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <title>ASSIGNMENT</title>
6 <link rel="stylesheet" href="{{url_for('redirect_to', link='https://s3.jp-fok.cloud-object-storage.appdomain.cloud/sanrasheedcss/assign3.css')}}" type="text/css">
7 </head>
8 <body>
9 <h1>DISPLAYING IMAGES FROM CLOUD OBJECT STORAGE</h1>
10 {% for image in arr: %}
11 
12 {% endfor %}
13 </body>
14 </html>
```

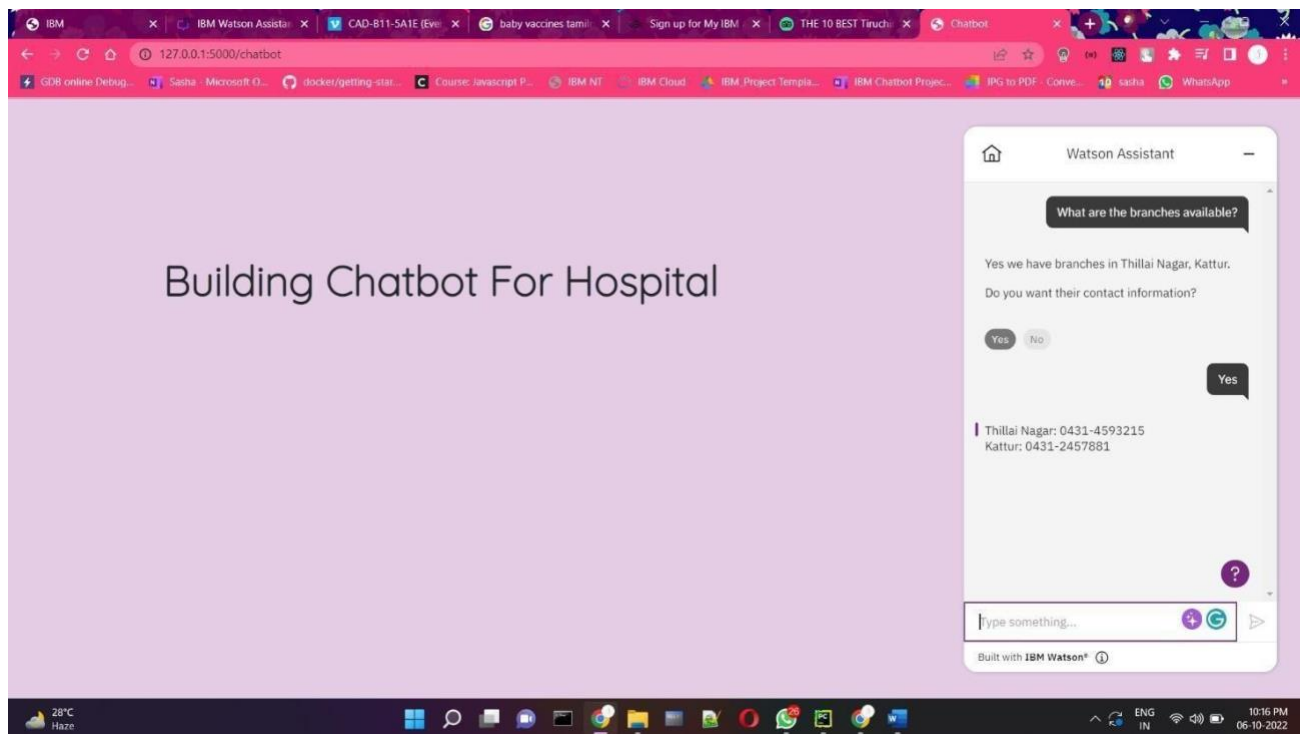
The same CSS file is used inside the HTML Code.

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.

A chatbot using IBM Watson assistant for hospital has been designed.



Now the user comes with query to know the branches for that hospital in the city.

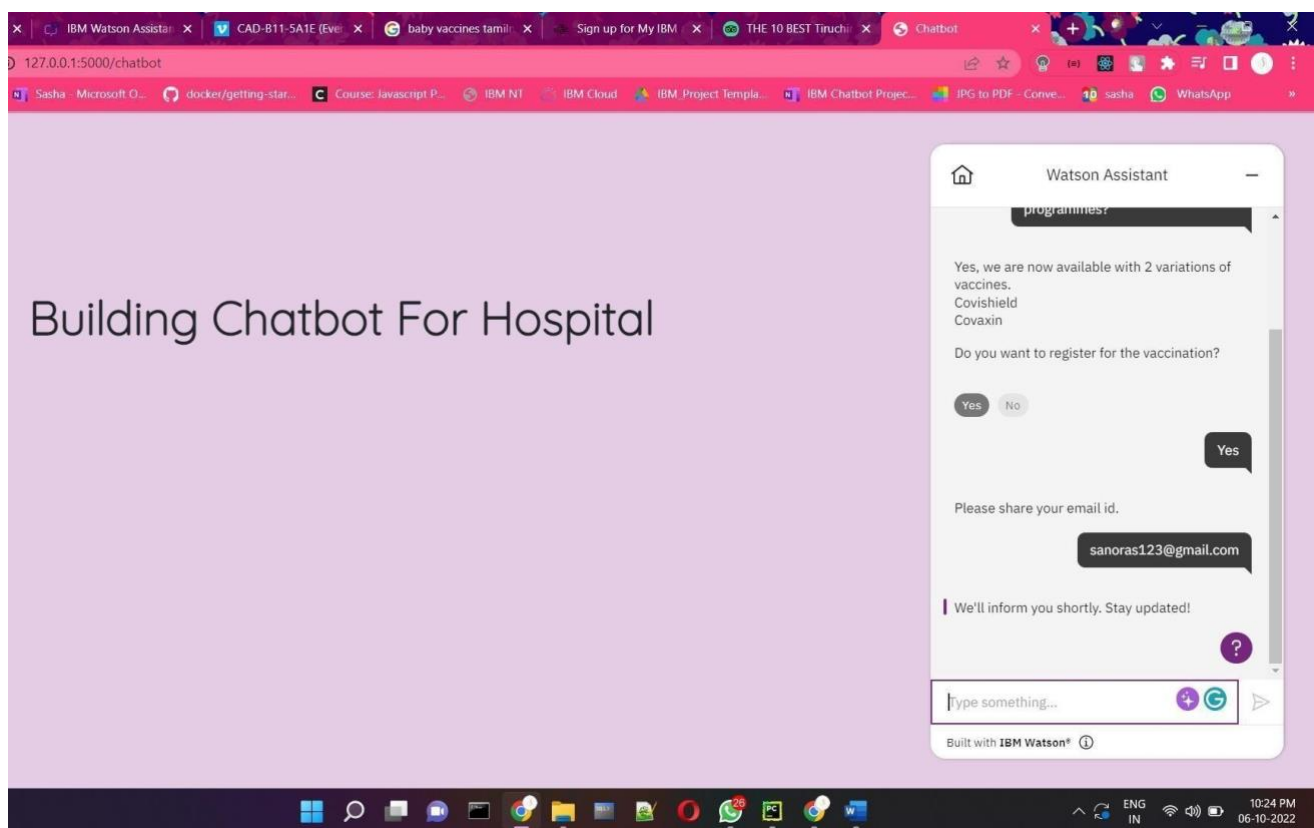
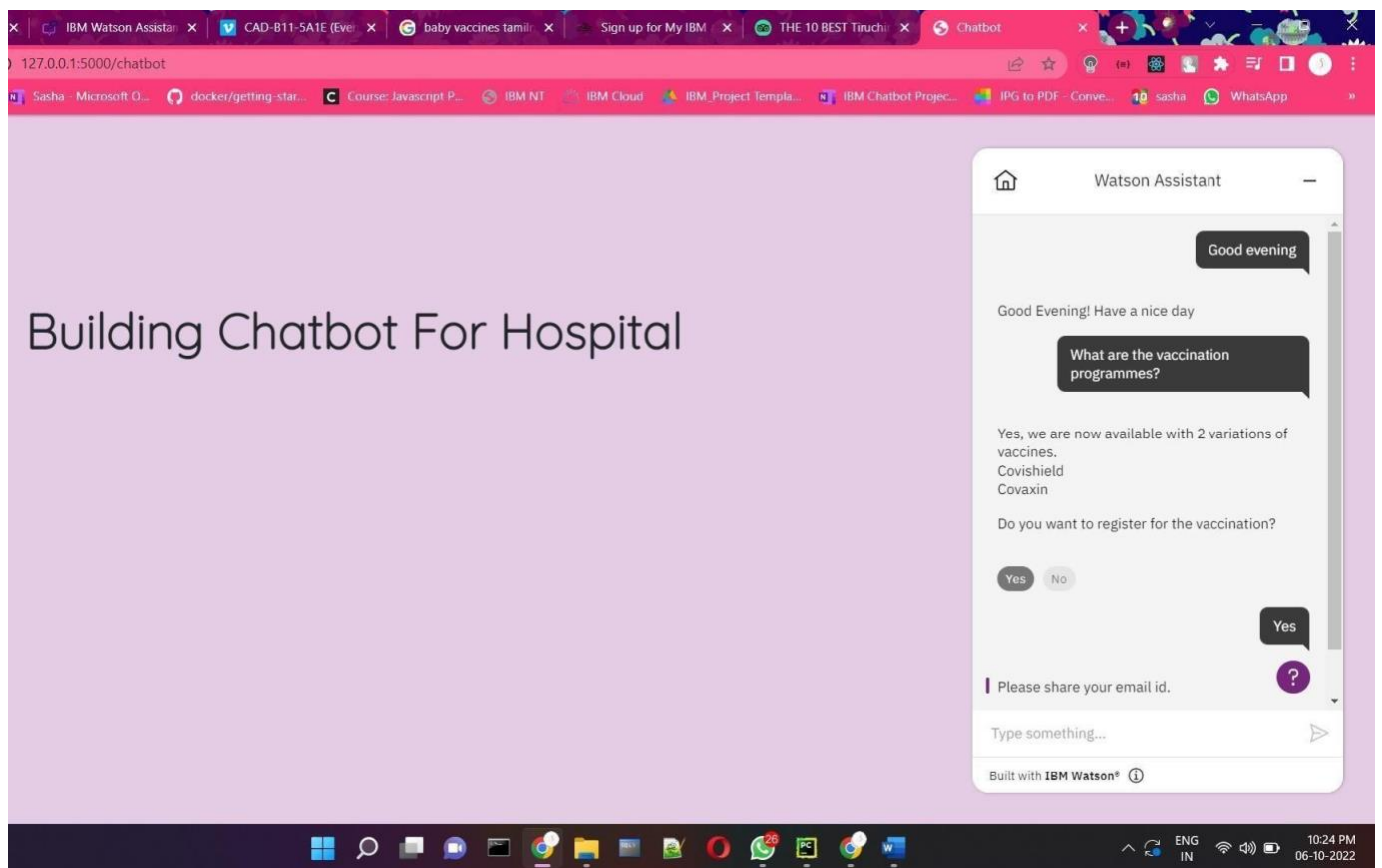


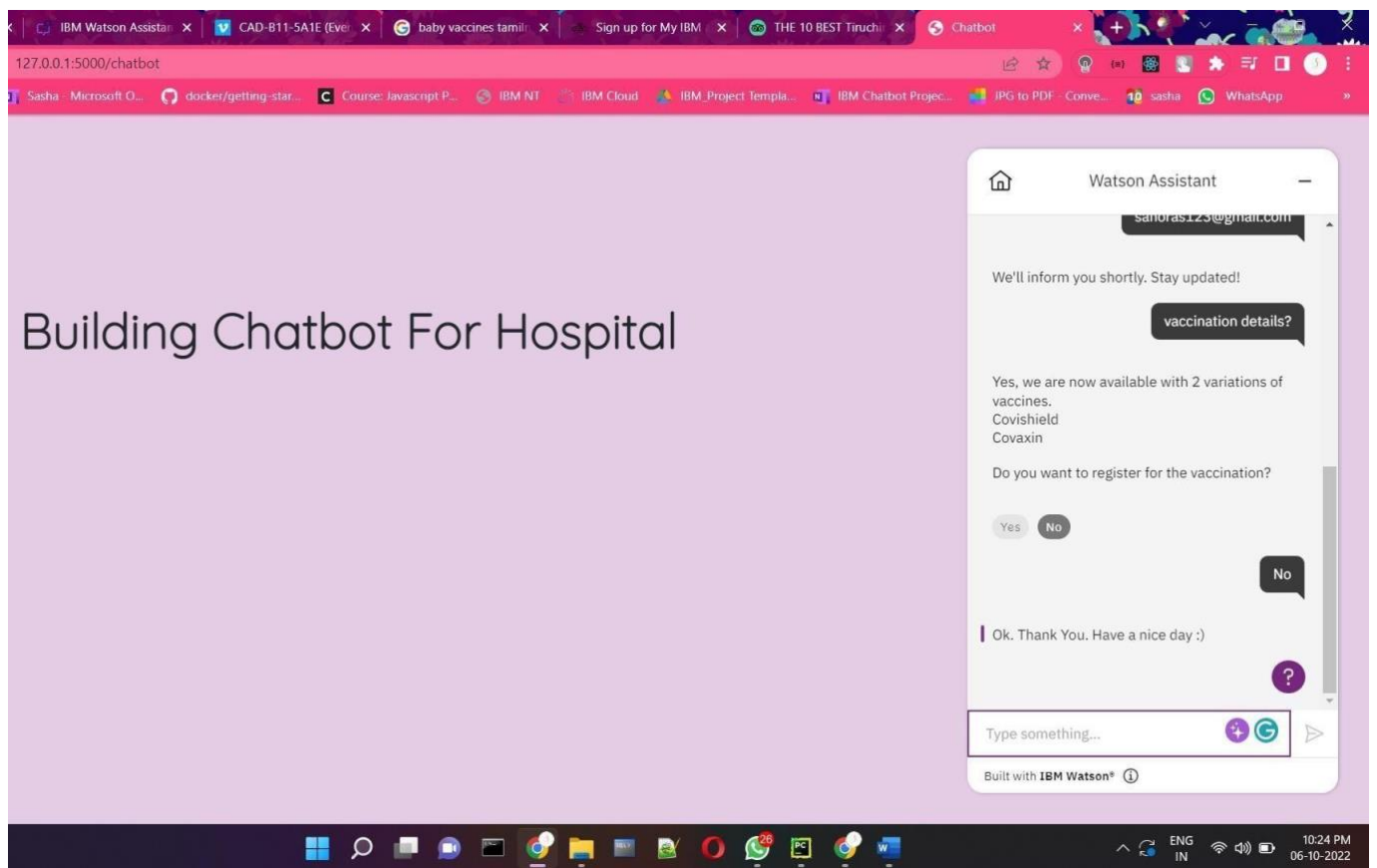
WEB URL OF ASSISTANT :

```
<script>
window.watsonAssistantChatOptions = {
  integrationID: "cba0c163-9f9a-42ce-b6e9-979ac6318cc0", // The ID of this integration.
  region: "au-syd", // The region your integration is hosted in.
  serviceInstanceID: "245858a6-01a2-41fc-8707-d603be17b8f0", // 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>
```

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

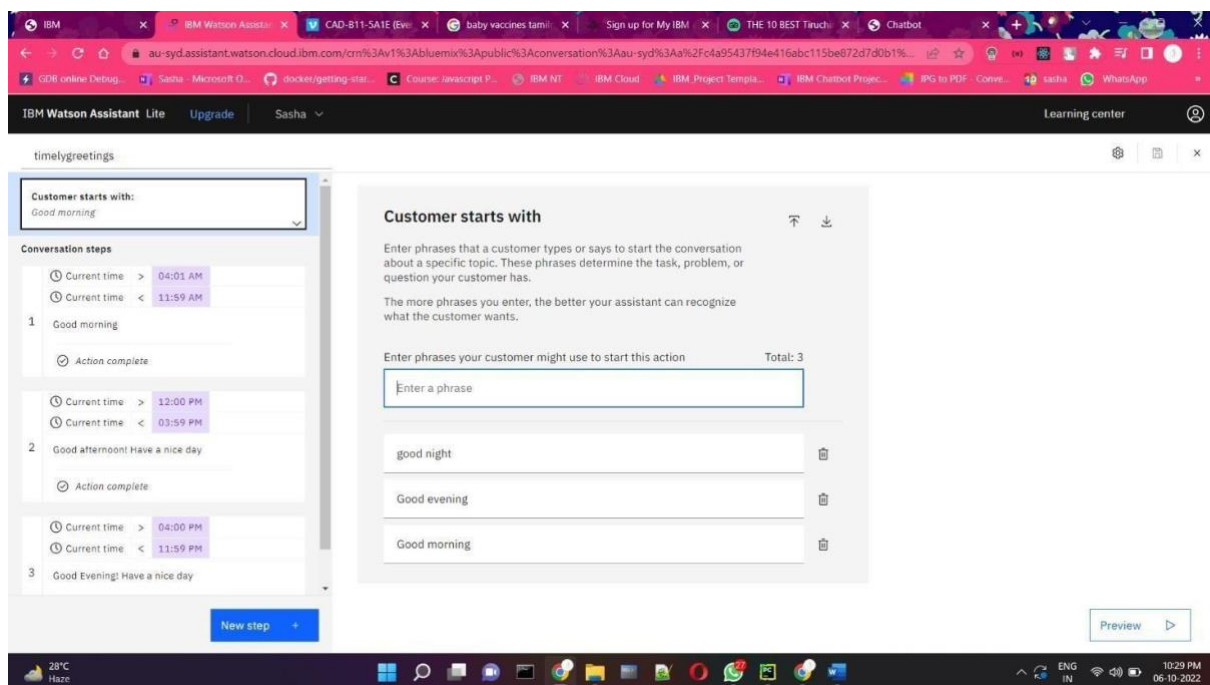
The user is coming up with a query of vaccination details.





A Watson assistant with 3 conditions is given.

GREETINGS ACTION:



BRANCHES QUERY ACTION:

The screenshot shows the IBM Watson Assistant interface for a project named 'Sasha'. The top navigation bar includes 'IBM Watson Assistant', 'Upgrade', and 'Sasha'. The main content area is titled 'branches' and features a 'Conversation steps' panel on the left and a 'Customer starts with' panel on the right.

Conversation steps:

- Step 1: 'Yes we have branches in Thiruvalluvar Nagar, Kattur.' with a 'Continue to next step' button.
- Step 2: 'Do you want their contact information?' with a 'Confirmation' button.
- Step 3: 'Thiruvalluvar Nagar: 0431-8593213 Kattur: 0431-2457881' with an 'Action complete' button.
- Step 4: 'Thank You :)' with an 'Action complete' button.

Customer starts with:

Enter phrases that a customer types or says to start the conversation about a specific topic. These phrases determine the task, problem, or question your customer has.

The more phrases you enter, the better your assistant can recognize what the customer wants.

Enter phrases your customer might use to start this action. Total: 5

- Enter a phrase
- does your hospital has any other branches in the city?
- is there any branch near me
- Branches please
- May i know your other branches in this city?
- Where are your branches?

A 'Preview' button is located at the bottom right of the 'Customer starts with' panel.

VACCINATION QUERY ACTION:

The screenshot shows the IBM Watson Assistant interface for a project named 'Sasha'. The top navigation bar includes 'IBM Watson Assistant', 'Upgrade', and 'Sasha'. The main content area is titled 'vaccination' and features a 'Conversation steps' panel on the left and a 'Customer starts with' panel on the right.

Conversation steps:

- Step 1: 'vaccines, Covishield, Covaxin' with a 'Continue to next step' button.
- Step 2: 'Do you want to register for the vaccination?' with a 'Confirmation' button.
- Step 3: 'Please share your email id.' with a 'Register' button.
- Step 4: 'We'll inform you shortly. Stay updated!' with an 'Action complete' button.

Customer starts with:

Enter phrases that a customer types or says to start the conversation about a specific topic. These phrases determine the task, problem, or question your customer has.

The more phrases you enter, the better your assistant can recognize what the customer wants.

Enter phrases your customer might use to start this action. Total: 2

- Enter a phrase
- is vaccination available for covid
- vaccination

A 'Preview' button is located at the bottom right of the 'Customer starts with' panel.

HTML CODE WITH ASSISTANT EMBEDDED:

```
<!DOCTYPE html>
<html><head>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Chatbot </title>
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/css/bootstrap.min.css" integrity="sha384-
xOolHFLEh07PJGoPkLv1IbcEPTNtaed2xpHsD9ESMhqIYd0nLMwNLD69Npy4HI+N"
crossorigin="anonymous">
  <link rel="stylesheet" href="{ {url_for('static',filename='styles/login.css')}}" type="text/css">
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <script defer src="https://use.fontawesome.com/releases/v5.13.1/js/all.js"></script>
  <link
href="https://fonts.googleapis.com/css2?family=Alex+Brush&family=Quicksand:wght@300&displ
ay=swap" rel="stylesheet">
  </head>
<body style="background:rgba(128,0,128,0.2)">
<h1 style="font-weight: bolder;
margin: 12%;
font-size: xxx-large;">Building Chatbot For Hospital</h1>
<script>
window.watsonAssistantChatOptions = {
  integrationID: "cba0c163-9f9a-42ce-b6e9-979ac6318cc0", // The ID of this integration.
  region: "au-syd", // The region your integration is hosted in.
  serviceInstanceID: "245858a6-01a2-41fc-8707-d603be17b8f0", // 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>
</body>
</html>
```