

# Image Recognition with IBM Cloud Visual Recognition

## PHASE\_3

Design a simple web interface where users can upload images and view the AI-generated captions.



## **1. Image Recognition Overview:**

- Image recognition, also known as computer vision, is a field of artificial intelligence that enables machines to interpret and understand the content of images and videos. IBM Cloud Visual Recognition is a cloud-based service that simplifies the process of building image recognition capabilities into applications.

## **2. Setting Up IBM Cloud Visual Recognition:**

- To create an image recognition system, you need an IBM Cloud account. After creating an account, you can set up the Visual Recognition service in the IBM Cloud platform. This service allows you to analyze and classify images using machine learning models.

## **3. Obtaining API Keys:**

- To access the Visual Recognition service, you need API keys that authenticate your requests. API keys are obtained when you create your Visual Recognition service instance in the IBM Cloud platform. These keys are essential for integrating the service into your application.

## **4. Web Interface Design:**

- A key component of your image recognition system is the user-facing web interface. The web

interface should be designed to provide users with the ability to upload images for analysis and view the AI-generated captions. Here's a breakdown of the web interface components:

- **HTML Structure:** HTML is used to define the structure of the web page, including input forms, buttons, and areas for displaying results.

- **Form for Image Upload:** You create an HTML form that includes an `

- **User Interaction:** JavaScript is used to handle user interactions. When an image is uploaded, JavaScript triggers a request to the Visual Recognition service for analysis.

- **Display of Results:** The web interface includes an area to display the AI-generated captions and may also include error messages for user feedback.

- **CSS Styling:** CSS is used to style the web page, making it visually appealing and user-friendly. You can customize fonts, colors, layout, and other visual elements.

## 5.Integration with IBM Cloud Visual Recognition:

- Within your Python code, you integrate the Visual Recognition service using the `ibm-watson` library. Key steps for integration include:

- **Authentication:** You use your API keys to create an authenticator and configure the Visual Recognition service instance with these credentials.

- **Image Classification:** The code sends the uploaded image to the Visual Recognition service for analysis. The service identifies and classifies objects in the image, providing AI-generated captions or labels.

## 6. Handling Errors:

- It's important to implement error handling within your code to provide feedback to users. For example, if the Visual Recognition service encounters issues with image analysis, it should return an error message to the web interface.

## 7. Displaying Results:

- Successful image analysis results are displayed on the web interface. The AI-generated captions or

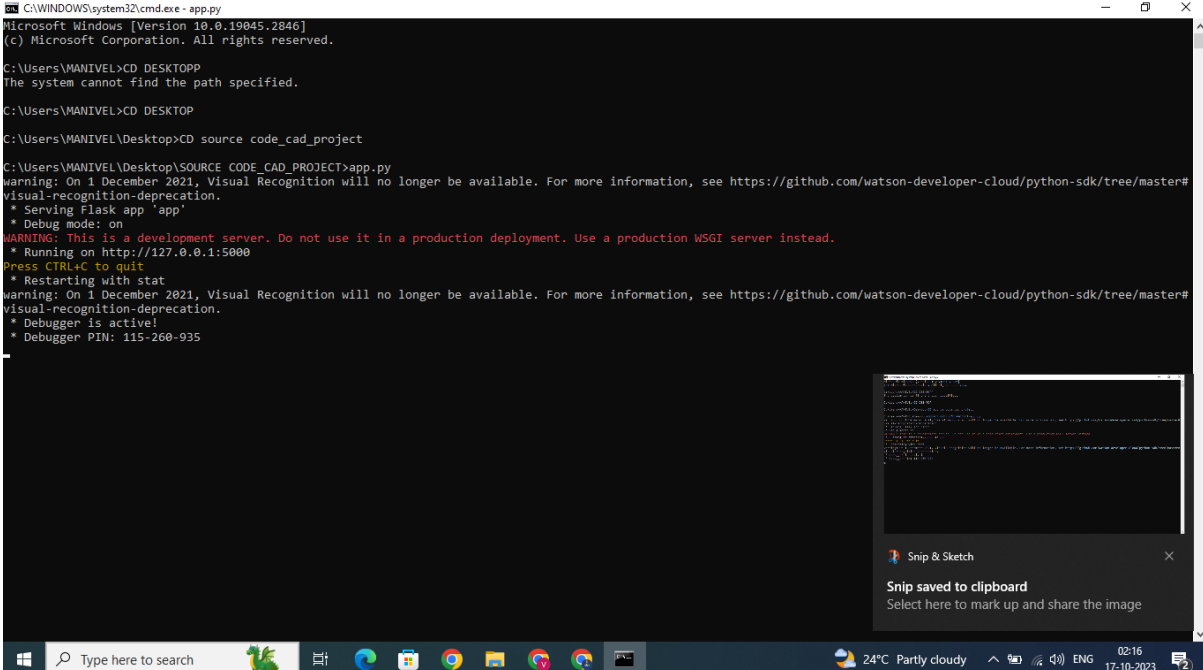
labels are shown to users, providing them with valuable insights into the content of the uploaded images.

## 8. Further Enhancements:

- To create a production-ready image recognition system, you can consider additional features such as user authentication, image storage, and more advanced user interfaces. Deployment on a web server or cloud platform is also necessary to make the system accessible online.

## OUTPUT

## Host Generation



```
C:\WINDOWS\system32\cmd.exe - app.py
Microsoft Windows [Version 10.0.19045.2846]
(c) Microsoft Corporation. All rights reserved.

C:\Users\MANIVEL>CD DESKTOP
The system cannot find the path specified.

C:\Users\MANIVEL>CD DESKTOP

C:\Users\MANIVEL\Desktop>CD source_code_cad_project

C:\Users\MANIVEL\Desktop\SOURCE_CODE_CAD_PROJECT>app.py
Warning: On 1 December 2021, Visual Recognition will no longer be available. For more information, see https://github.com/watson-developer-cloud/python-sdk/tree/master#visual-recognition-deprecation.
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
Warning: On 1 December 2021, Visual Recognition will no longer be available. For more information, see https://github.com/watson-developer-cloud/python-sdk/tree/master#visual-recognition-deprecation.
 * Debugger is active!
 * Debugger PIN: 115-260-935
```

# IMAGE CAPTION GENERATOR WEB INTERFACE



## Source code:

### HTML Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport"
content="width=device-width, initial-scale=1.0">
  <title>Image Recognition</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      text-align: center;
```

```
}  
.container {  
    max-width: 500px;  
    margin: 0 auto;  
    padding: 20px;  
}  
h1 {  
    font-size: 24px;  
}  
form {  
    margin-top: 20px;  
}  
input[type="file"] {  
    width: 100%;  
    padding: 10px;  
    border: 1px solid #ccc;  
    border-radius: 5px;  
}  
input[type="submit"] {  
    background-color: #007BFF;  
    color: #fff;  
    border: none;  
    padding: 10px 20px;  
    border-radius: 5px;  
    cursor: pointer;  
}  
p {
```

```

        margin-top: 20px;
        font-weight: bold;
    }
    .error {
        color: red;
    }
</style>
</head>
<body>
    <div class="container">
        <h1>Image Recognition</h1>
        <form    method="POST"
enctype="multipart/form-data">
            <input type="file" name="image"
accept="image/*" required>
            <input type="submit" value="Upload and
Analyze">
        </form>
        {% if error %}
        <p class="error">{{ error }}</p>
        {% endif %}
        {% if caption %}
        <p>{{ caption }}</p>
        {% endif %}
    </div>
</body>
</html>

```



## FLASK Code:(app.py)

```
from flask import Flask, render_template, request
from ibm_watson import VisualRecognitionV4
from ibm_watson.visual_recognition_v import
FileWithMetadata
from ibm_cloud_sdk_core.authenticators import
IAMAuthenticator

app = Flask(__name)

# Replace with your actual API key and service URL
api_key = "YOUR_API_KEY"
service_url = "YOUR_SERVICE_URL"

authenticator = IAMAuthenticator(api_key)
visual_recognition =
VisualRecognitionV4(version="2022-08-20",
authenticator=authenticator
)
visual_recognition.set_service_url(service_url)

@app.route("/", methods=["GET", "POST"])
def index():
    caption = None
    if request.method == "POST":
        if "image" not in request.files:
            return render_template("index.html", error="No
file part")
```

```

image = request.files["image"]

if image.filename == "":
    return render_template("index.html", error="No
selected file")

try:
    # Use Visual Recognition to analyze the
uploaded image
    response = visual_recognition.analyze(
        collection_ids=["your_collection_id"],
        features=["objects"],
        images_file=image
    ).get_result()

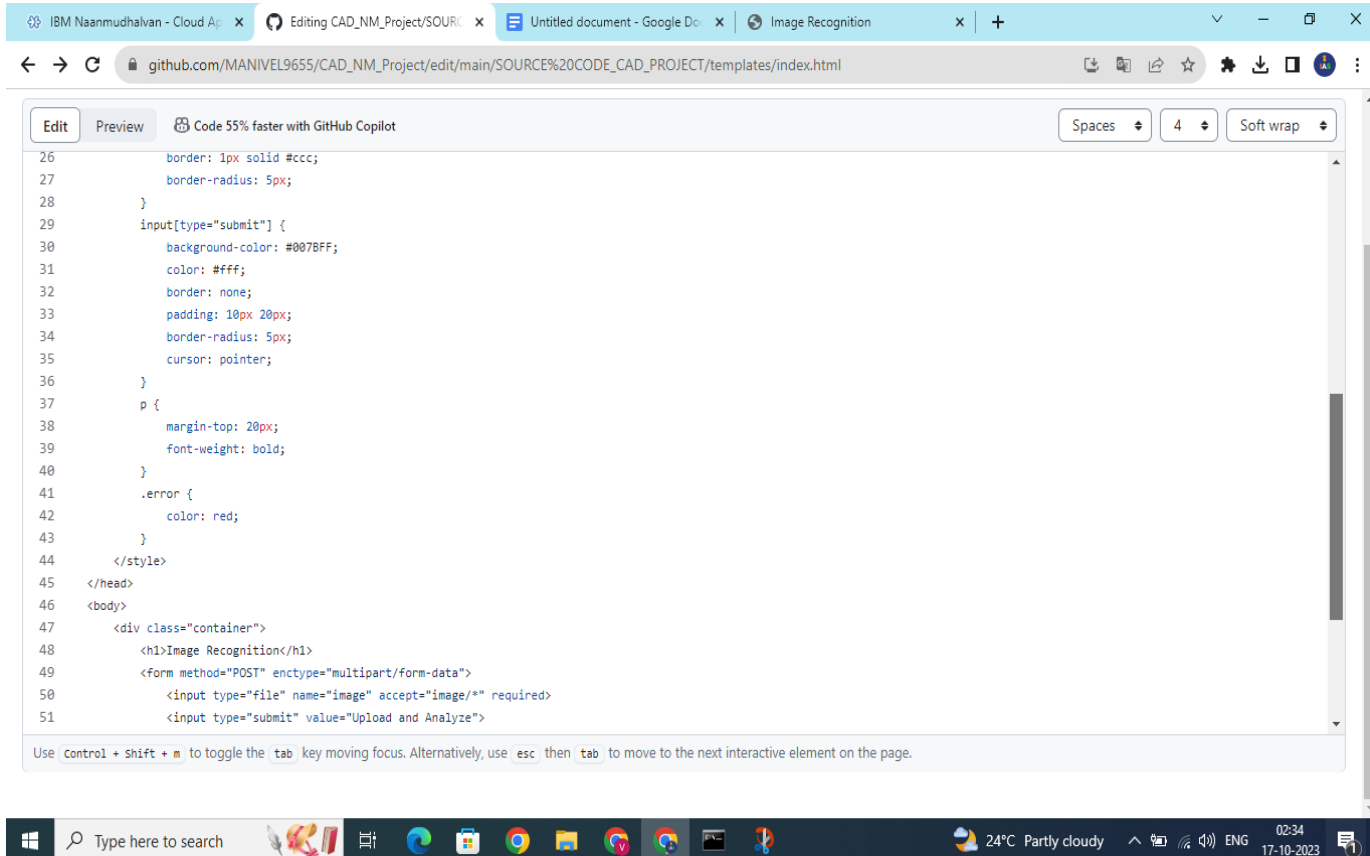
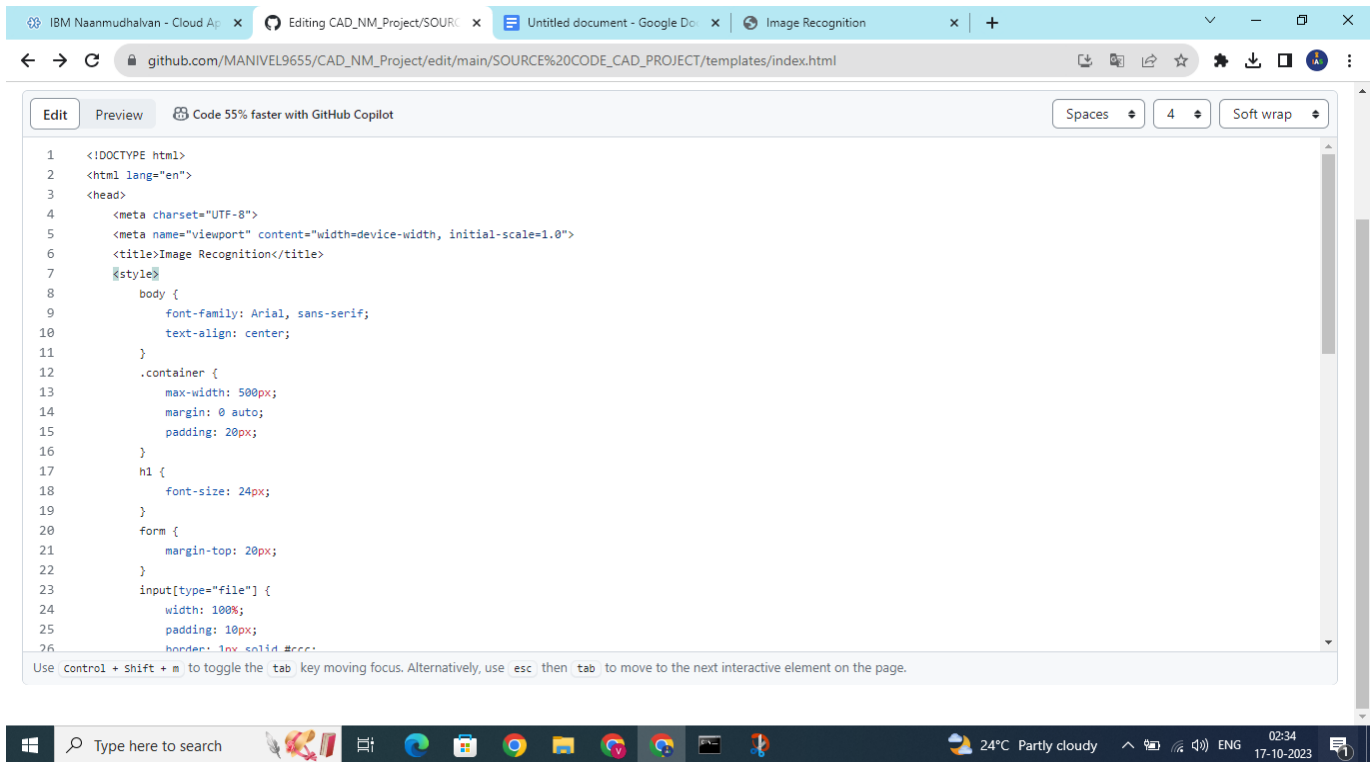
    # Extract the top caption from the response
    top_class =
response["images"][0]["classifiers"][0]["classes"][0]["clas
s"]

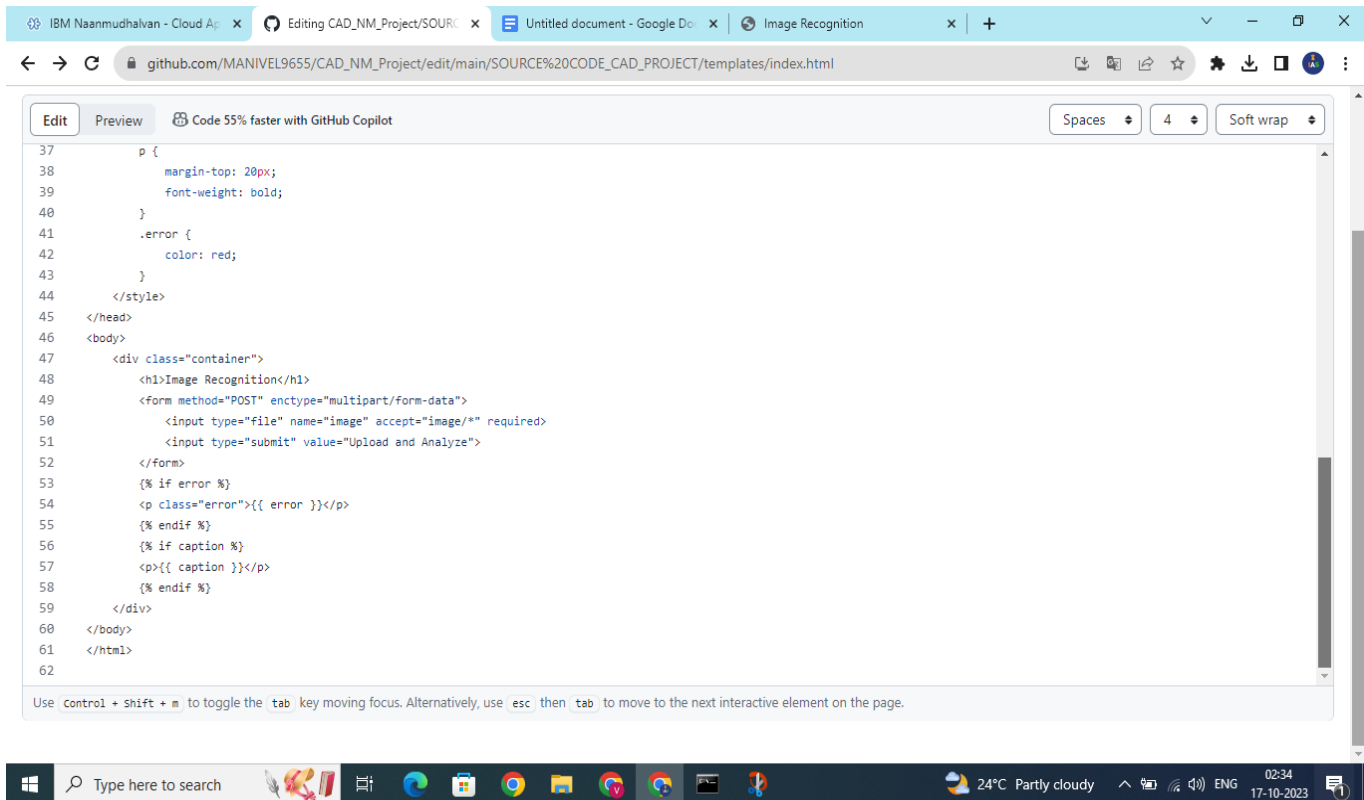
    caption = f"AI-generated Caption: {top_class}"
except Exception as e:
    return render_template("index.html", error=str(e))

return render_template("index.html", caption=caption)

if __name__ == "__main__":
    app.run(debug=True)

```





IBM Naanmudhalvan - Cloud Ap x CAD\_NM\_Project/SOURCE CODE x Untitled document - Google Do x Image Recognition x +

github.com/MANIVEL9655/CAD\_NM\_Project/blob/main/SOURCE%20CODE\_CAD\_PROJECT/app.py

Code Blame 49 lines (38 loc) · 1.65 KB Code 55% faster with GitHub Copilot Raw

```
1 # Install necessary libraries: pip install Flask ibm-watson ibm-cloud-sdk-core
2
3 from flask import Flask, render_template, request
4 from ibm_watson import VisualRecognitionV4
5 from ibm_watson.visual_recognition_v4 import FileWithMetadata
6 from ibm_cloud_sdk_core.authenticators import IAMAuthenticator
7
8 app = Flask(__name__)
9
10 # Replace with your actual API key and service URL
11 api_key = "YOUR_API_KEY"
12 service_url = "YOUR_SERVICE_URL"
13
14 authenticator = IAMAuthenticator(api_key)
15 visual_recognition = VisualRecognitionV4(
16     version="2022-08-20",
17     authenticator=authenticator
18 )
19 visual_recognition.set_service_url(service_url)
20
21 @app.route("/", methods=["GET", "POST"])
22 def index():
23     caption = None
24     if request.method == "POST":
25         # Handle image upload
26         if "image" not in request.files:
27             return render_template("index.html", error="No file part")
28
29         image = request.files["image"]
30
31         if image.filename == "":
32             return render_template("index.html", error="No selected file")
33
34         try:
35             # Use Visual Recognition to analyze the uploaded image
36             response = visual_recognition.classify(
37                 images_file=FileWithMetadata(image)
38             ).get_result()
39
40             # Extract the top caption from the response
41             top_class = response["images"][0]["classifiers"][0]["classes"][0]["class"]
42             caption = f"AI-generated Caption: {top_class}"
43         except Exception as e:
44             return render_template("index.html", error=str(e))
45
46     return render_template("index.html", caption=caption)
47
48 if __name__ == "__main__":
49     app.run(debug=True)
```

Type here to search 24°C Partly cloudy 02:35 17-10-2023

IBM Naanmudhalvan - Cloud Ap x CAD\_NM\_Project/SOURCE CODE x Untitled document - Google Do x Image Recognition x +

github.com/MANIVEL9655/CAD\_NM\_Project/blob/main/SOURCE%20CODE\_CAD\_PROJECT/app.py

main CAD\_NM\_Project / SOURCE CODE\_CAD\_PROJECT / app.py ↑ Top

Code Blame 49 lines (38 loc) · 1.65 KB Code 55% faster with GitHub Copilot Raw

```
22 def index():
23
24     if "image" not in request.files:
25         return render_template("index.html", error="No file part")
26
27     image = request.files["image"]
28
29     if image.filename == "":
30         return render_template("index.html", error="No selected file")
31
32     try:
33         # Use Visual Recognition to analyze the uploaded image
34         response = visual_recognition.classify(
35             images_file=FileWithMetadata(image)
36         ).get_result()
37
38         # Extract the top caption from the response
39         top_class = response["images"][0]["classifiers"][0]["classes"][0]["class"]
40         caption = f"AI-generated Caption: {top_class}"
41     except Exception as e:
42         return render_template("index.html", error=str(e))
43
44     return render_template("index.html", caption=caption)
45
46 if __name__ == "__main__":
47     app.run(debug=True)
```

Type here to search 24°C Partly cloudy 02:35 17-10-2023