**PHASE 4 ASSIGNMENT**

**PROBLEM STATEMENT:**

Continue building the image recognition system by integrating IBM Cloud Visual Recognition and AI-generated captions. Implement the image classification process using the IBM Cloud Visual Recognition API.

**Use natural language generation to create captions for the recognized images.**

**Aim:**

To implement the image classification process using the IBM Cloud Visual Recognition API and to use natural language generation to create captions for the recognized images.

**Setting Up IBM Cloud Visual Recognition API:**

To integrate IBM Cloud Visual Recognition into your project, you'll need to make API requests to classify images. You can use the Visual Recognition service API to achieve this. Here's a basic overview of how to do it:

Use your Visual Recognition API key and endpoint, which you can find in your IBM Cloud Visual Recognition service instance.

Make API calls to classify images by sending them to the Visual Recognition service.

Here's a simplified example in Python using the requests library:

import requests

api\_key = 'your\_api\_key'

endpoint = 'your\_endpoint'

image\_url = 'url\_to\_your\_image.jpg'

headers = {

'apikey': api\_key,

}

params = {

'url': image\_url,

}

response = requests.get(f'{endpoint}/v3/classify', headers=headers, params=params)

if response.status\_code == 200:

data = response.json()

# Process the classification results

else:

print('Error:', response.status\_code)

**Implementing Natural Language Generation (NLG):**

To generate captions for recognized images, you can use NLG techniques. One popular approach is to use pre-trained models like GPT (Generative Pre-trained Transformer) or OpenAI's GPT-3 to generate descriptive text based on the image classification results.

Here's a simplified example using OpenAI's GPT-3 API in Python:

import openai

openai.api\_key = 'your\_openai\_api\_key'

def generate\_caption(classification\_results):

# Convert classification results to a descriptive prompt

prompt = f"Describe the image: {classification\_results['images'][0]['class']}"

response = openai.Completion.create(

engine="davinci",

prompt=prompt,

max\_tokens=50, # Adjust the max tokens as needed

)

caption = response.choices[0].text

return caption

# Pass your Visual Recognition results to the generate\_caption function

classification\_results = { ... } # Your Visual Recognition results

caption = generate\_caption(classification\_results)

print(caption)

**Integrate Image Classification and Caption Generation:**

You should call the Visual Recognition API to classify the image and then use the classification results as input to the caption generation function. This can be done within the same script or integrated into your project's backend or frontend.

**Displaying Captions:**

After generating captions, you can display them alongside the recognized images in your application. You may want to use a user-friendly interface to make the captions easily readable and accessible.

**Optimizing and Scaling:**

As your project scales, consider optimizing your image recognition and caption generation processes for speed and resource efficiency. You may also need to manage your API call limits and costs.

**Monitoring and Improvement:**

Continuously monitor the performance of your image recognition and caption generation. Collect user feedback and fine-tune the AI models as needed to improve the quality of generated captions.