**Image Recognition with IBM Cloud Visual Recognition**

**Phase 4**

To build an image recognition system by integrating IBM Cloud Visual Recognition and AI-generated captions, you can follow these steps:

Create an IBM Cloud account and enable the Visual Recognition service.

1.Obtain an API key for the Visual Recognition service.

2.Install a natural language generation (NLG) library on your machine.

3.Write a program to perform the following steps:

4.Upload an image to IBM Cloud Object Storage.

• Use the Visual Recognition API to classify the image.

• Use the NLG library to generate a caption for the recognized image.

• Display the image and the generated caption to the user.

**Code**:

import io

import json

import requests

from ibm\_cloud\_sdk\_core.authenticators import IAMAuthenticator

from ibm\_watson import VisualRecognitionV3

# Create an IAM authenticator

authenticator = IAMAuthenticator('YOUR\_API\_KEY')

# Create a Visual Recognition client

visual\_recognition = VisualRecognitionV3(authenticator)

# Classify an image

def classify\_image(image\_path):

with open(image\_path, 'rb') as image\_file:

image\_data = image\_file.read()

response = visual\_recognition.classify(images\_file=io.BytesIO(image\_data))

# Get the classification results

classification\_results = json.loads(response.content)['images'][0]['classifiers'][0]['classes']

return classification\_results

# Generate a caption for an image

def generate\_caption(classification\_results):

# TODO: Implement this function using an NLG library

# For example, you could use the following pseudocode:

# caption = ''

# for class\_result in classification\_results:

# caption += class\_result['name'] + ' '

#

# return caption

# Display the image and the generated caption

def display\_image\_and\_caption(image\_path, caption):

# TODO: Implement this function using a GUI library

# For example, you could use the following pseudocode:

# image = Image.open(image\_path)

# image.show()

# print(caption)

# Main function

def main():

image\_path = 'path/to/image.jpg'

# Classify the image

classification\_results = classify\_image(image\_path)

# Generate a caption for the image

caption = generate\_caption(classification\_results)

# Display the image and the generated caption

display\_image\_and\_caption(image\_path, caption)

if \_\_name\_\_ == '\_\_main\_\_':

main()

**Explanation of program:**

The program first creates an IAM authenticator and a Visual Recognition client. Then, it defines two functions:

• classify\_image(): This function takes the path to an image as input and returns a list of objects detected in the image, along with their scores.

• generate\_caption(): This function takes the results of the image classification as input and generates a caption for the image.

The main function of the program then calls the classify\_image() function to classify the image specified by the image\_path variable. It then calls the generate\_caption() function to generate a caption for the image. Finally, it calls the display\_image\_and\_caption() function to display the image and the caption.

The generate\_caption() function is currently implemented as a TODO. You will need to implement this function using a natural language generation (NLG) library. For example, you could use the Google Cloud Natural Language API or the Amazon Rekognition API.

Once you have implemented the generate\_caption() function, you can run the program to classify images and generate captions for them. To do this, simply specify the path to the image you want to classify as the value of the image\_path variable.

The program will then display the image and the generated caption.

Once you have implemented the image recognition system, you can use it to classify and caption images in a variety of applications. For example, you could use the system to:

• Develop a photo search engine that can search for images based on their content.

• Create a social media app that can automatically generate captions for uploaded images.

• Build an assistive technology tool that can help blind or visually impaired people to understand the content