**IBM Naan Mudhalvan**

**Cloud Application Development**

**(Image recognition and IBM**

**Cloud visual recognition)**

**Phase-4 Project**

**Image caption generating**

|  |
| --- |
| import urllib  import requests  import os   # retrieving using image url  urllib.request.urlretrieve("https://i.ibb.co/xY4DJJ5/img1.jpg", "img1.jpg")  urllib.request.urlretrieve("https://i.ibb.co/Gnd1Y1L/img2.jpg", "img2.jpg")  urllib.request.urlretrieve("https://i.ibb.co/Z6JgS1L/img3.jpg", "img3.jpg")   print('Images downloaded')   # get current working directory path  path = os.getcwd()  captionarr = [      "This is the first caption",      "This is the second caption",      "This is the third caption"      ]  # importing necessary functions from PIL  from PIL import Image  from PIL import ImageFont  from PIL import ImageDraw   # print(os.getcwd())   # checking the file mime types if  # it is jpg, png or jpeg  def ext(file):      index = file.find(".jpg")      current\_file = ""      current\_file = file[index:]      return current\_file   def ext2(file):      index = file.find(".jpeg")      current\_file = ""      current\_file = file[index:]      return current\_file  def ext3(file):      index = file.find(".png")      current\_file = ""      current\_file = file[index:]      return current\_file   # converting text from lowercase to uppercase  def convert(words):      s = ""      for word in words:          s += word.upper()      return s   caption\_first = convert(captionarr[0])  caption\_second = convert(captionarr[1])  caption\_third = convert(captionarr[2])  print(caption\_first)  print(caption\_second)  print(caption\_third)  count = 0   for f in os.listdir('.'):      try:          # Checking for file types if jpg, png          # or jpeg excluding other files          if (ext(f) == '.jpg' or ext2(f) == '.jpeg' or ext3(f) == '.png'):              img = Image.open(f)              width, height = img.size              basewidth = 1200              # print(height)   # Resizing images to same width height              wpercent = (basewidth / float(img.size[0]))              hsize = int((float(img.size[1])\*float(wpercent)))              img = img.resize((basewidth, hsize), Image.ANTIALIAS)              new\_width, new\_height = img.size              # print(new\_height)              # changing image mode if not in RGB              if not img.mode == 'RGB':                  img = img.convert('RGB')                draw = ImageDraw.Draw(img)              # font = ImageFont.truetype(<font-file>, <font-size>)              # initializing which font will be chosen by us              font = ImageFont.truetype("Arial Bold.ttf", 35)                 # First Caption on First image              if count == 0:                  draw.text((new\_width / 15 + 25, new\_height - 100),                             caption\_first, (255, 0, 0), font = font,                             align ="center")                # Second Caption on Second image              elif count == 1:                  draw.text((new\_width / 15 + 25, new\_height - 100),                            caption\_second, (255, 0, 0), font = font,                            align ="center")                # Third Caption on Third image              else:                  draw.text(( new\_width / 15 + 25, new\_height - 100),                              caption\_third, (255, 0, 0), font = font,                              align ="center")  img.save("CaptionedImges/{}".format(f))              print('done')              count = count + 1        except OSError:          pass  import os  import glob  import shutil    # changing directory to CaptionedImages  os.chdir(".\\CaptionedImges")    fnames = []  for file in os.listdir('.'):      # appending files in directory to the frames arr      fnames.append(file)    # sorting the files in frames array  # on the basis of last modified time  # reverse = True means ascending order sorting  fnames.sort(key = lambda x: os.stat(x).st\_ctime, reverse = True) |

**Step #2:**

Python3

|  |
| --- |
|  |

**Step #3:**Sorting the output files in accordance to last modified time so that they do not get placed in alphabetical or any other mismanaged order.

Python3

|  |
| --- |
|  |

**Output:**