AI/ML Developer Practical Test: OCR Application for ID Card Data Extraction:

Please find the below detail of my project:

Libraries Used:

- **pip install googletrans==4.0.0-rc1**
- **from googletrans import Translator**
- **import spacy**
- **import re**
- **from datetime import datetime**
- **import pandas as pd**
- **from tabulate import tabulate**
- **import cv2**
- **import numpy as np**
- **import matplotlib.pyplot as plt**

File Names:

- **ID Card name: Identity.jpg**
- **OCR processed text file : Identity_ocr.txt**
- **Translated text file : Translated.txt**
- **Output file : output.txt**

1.OCR Processing: As my current system not supporting to install Tesseract_OCR(faced source code issue), I have done the process for conversion through online. Here I have mentioned the code to be used for conversion

Below is the ID card I have taken to process: Image name is **Identity**

pip install pytesseract Pillow

from PIL import Image

import pytesseract

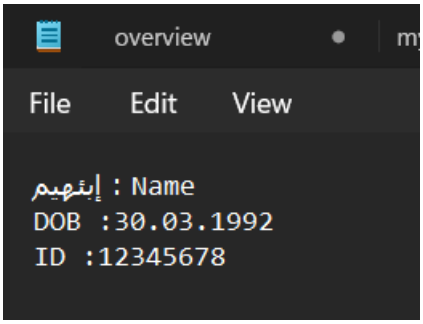
```
img = Image.open('path to the image/Identity.jpeg or png')

text= pytesseract.image_to_string(img)

with open('Identity.txt', 'w') as file:

    file.write(text)
```

2. Data Extraction and Structuring:



Above is the extracted data from the ID card.

If the same type of ID card is repeating means below program will help us

```
import nltk

from nltk.tokenize import word_tokenize

nltk.download('punkt')


tokens = word_tokenize(file_contents1)

data1 = [tokens[0]]

print(data1)

data2 = [tokens[5]]

print(data2)

data3 =[tokens[8]]

print(data3)
```

By tokening each word using nltk , we can save the contents which we need and store it in a variable to form data frame of table

PracticalTest.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Files

..
sample_data
Identity_ocr.txt
Translated.txt

Disk 81.39 GB available

+ Code + Text

```
pip install googletrans==4.0.0-rc1
```

```
[7] # Identity_ocr.txt is text file of image Identity
    with open("Identity_ocr.txt", "r") as file: # To open text file
        file_contents = file.read() # To read the file

    #pip install googletrans==4.0.0-rc1 -Please use this if installation needed
    from googletrans import Translator # using translator as it has arabic language
    # Translating Arabic text into English
    translator = Translator()
    text_to_translate = file_contents
    translated_text = translator.translate(text_to_translate, src='ar', dest='en')
    # Transfer translated text into new document Tranlated.txt
    A= translated_text.text
    with open('Translated.txt', 'w') as file:
         file.write(A)
    #Reading the Transalated text file
    with open("Translated.txt", "r") as file1:
        file_contents1 = file1.read()


    import spacy # importing spacy to extract person name from the text
    nlp = spacy.load("en_core_web_sm")
    # Process the text with SpaCy
    doc = nlp(file_contents1)
    # Save person names in a list for framing purpose
    person_names = [ent.text for ent in doc.ents if ent.label_ == "PERSON"]
```

```
[7] import re # importing re to extract date, ID number
    from datetime import datetime
    # searching date from the given string
    match_str = re.search(r'\d{2}.\d{2}.\d{4}', file_contents1)
    # check if any match is there
    if match_str:
        res = datetime.strptime(match_str.group(), '%d.%m.%Y').date()
    else:
        print("Nil")
    d2=([str(res)])#saving the date in variable d2


    #searching any ID number there in the file with 8 digit using RE
    eight_digit_numbers = re.findall(r'\b\d{8}\b', file_contents1)
    #saved the 8digit number in variable d3
    d3 = eight_digit_numbers


    import pandas as pd # importing pandas for Dataframing purpose
    from tabulate import tabulate # to make the data in table
    Frame = {"Name":person_names, "DOB":d2, "ID No":d3} # Framing the data under column name
    D = pd.DataFrame(Frame) #saving the data frame in variable D
    print(tabulate(D, headers = 'keys', tablefmt = 'psql'))# printing the data as a table
```

```
+----+--------+------------+----------+
|    | Name   | DOB        |    ID No |
|----+--------+------------+----------|
|  0 | Abaham | 1992-03-30 | 12345678 |
+----+--------+------------+----------+
```

Output:

```
+----+--------+------------+----------+
|    | Name   | DOB        |    ID No |
|----+--------+------------+----------|
|  0 | Abaham | 1992-03-30 | 12345678 |
+----+--------+------------+----------+
```

3. Image Handling:

I have used below libraries for image resizing and binarization

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

below functions also can be performed (for example)

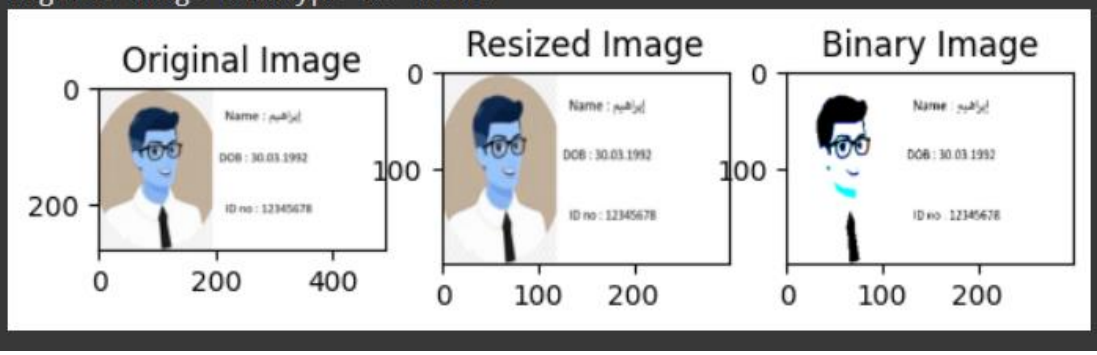
gray_img = ImageOps.grayscale(img) #converting image from other color shades_Gray scaling

threshold_img = = gray_img.point(lambda p: p > 127 and 256) # helps to seperate foreground and background

blurred_img = cv2.GaussianBlur(cv2.cvtColor(threshold_img, cv2.COLOR_GRAY2BGR), (4, 4), 0)# It helps in smoothing the image.

Output:

```
Orginal Image Shape is: (275, 489, 3)
Orginal Image Datatype is: uint8
```



Please find the program below:

```python
import cv2
import numpy as np
import matplotlib.pyplot as plt

img = cv2.imread("Identity.jpg")
print("Orginal Image Shape is:", img.shape)
print("Orginal Image Datatype is:",img.dtype)

#showing orginal image
plt.subplot(1, 3, 1)
plt.imshow(img, cmap='gray')
plt.title('Original Image')

resized_image = cv2.resize(img, (300, 200)) #changing width and height as per need
plt.subplot(1, 3, 2)# Display the resized image
plt.imshow(resized_image, cmap='gray')
plt.title('Resized Image')

# Binarization using adaptive thresholding
_, binary_image = cv2.threshold(resized_image, 128, 255, cv2.THRESH_BINARY)
plt.subplot(1, 3, 3)# Display the binary image
plt.imshow(binary_image, cmap='gray')
plt.title('Binary Image')

plt.show()
```