NAME: **Rufina M**

REG.NO. : 22MIC0019

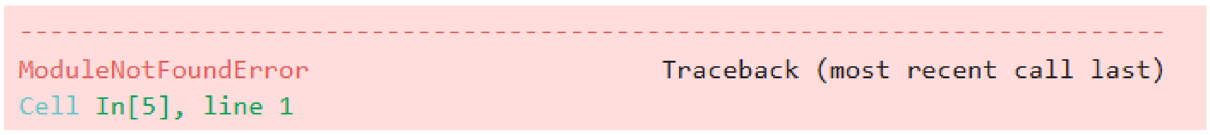
SLOT: **L7 + L8 + L27 + L28**

ADVANCED PYTHON PROGRAMMING LAB

TASK – 1

ModuleNotFound error is usually encountered when we directly try to import a library without installing them first.

I encountered the below error even after I tried importing pdfplumber once it's installed.



```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
Cell In[5], line 1
```

Code I gave:

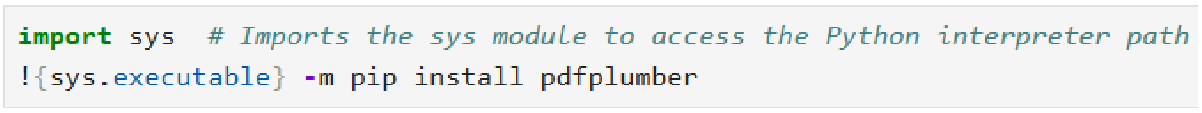
Cell1: !pip install pdfplumber

Cell2: import pdfplumber

REASON:

It is found that python interpreter has been installed in C drive and I wanted to keep my .ipynb files in D drive.

FIX:



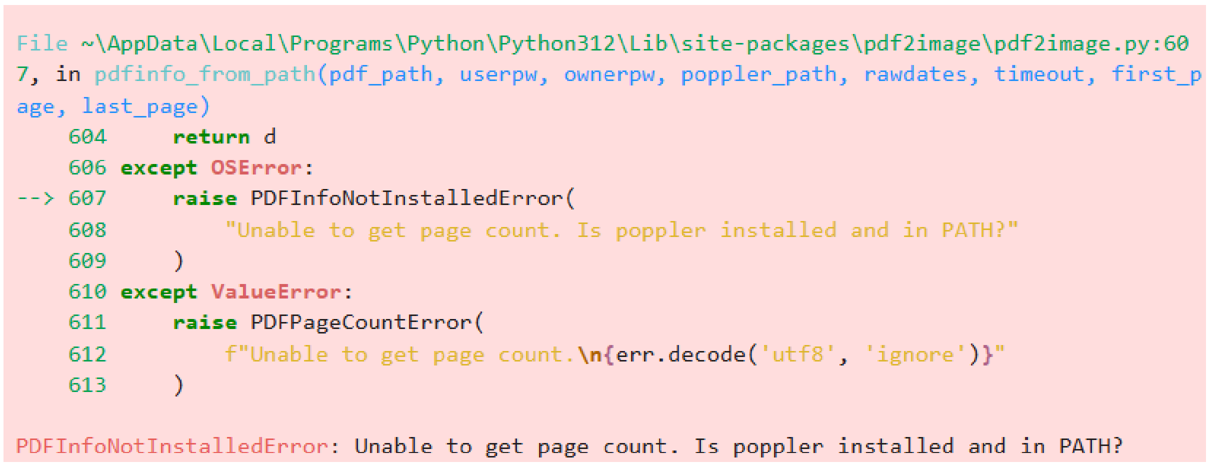
```
import sys  # Imports the sys module to access the Python interpreter path
!{sys.executable} -m pip install pdfplumber
```

{sys.executable} gives **the full path to the Python interpreter** that is currently running the notebook or script.

FileNotFound Error:

Cause: I haven't added poppler path correctly.

images = convert_from_path("sample_22mic0019.pdf", dpi=300)   #poppler_path should be the 3rd parameter.



```
File ~\AppData\Local\Programs\Python\Python312\Lib\site-packages\pdf2image\pdf2image.py:60
7, in pdfinfo_from_path(pdf_path, userpw, ownerpw, poppler_path, rawdates, timeout, first_p
age, last_page)
    604     return d
    606 except OSError:
--> 607     raise PDFInfoNotInstalledError(
    608         "Unable to get page count. Is poppler installed and in PATH?"
    609     )
    610 except ValueError:
    611     raise PDFPageCountError(
    612         f"Unable to get page count.\n{err.decode('utf8', 'ignore')}"
    613     )

PDFInfoNotInstalledError: Unable to get page count. Is poppler installed and in PATH?
```

FIX:

# The function convert_from_path() internally uses PIL.Image to create image objects from each page of the PDF.

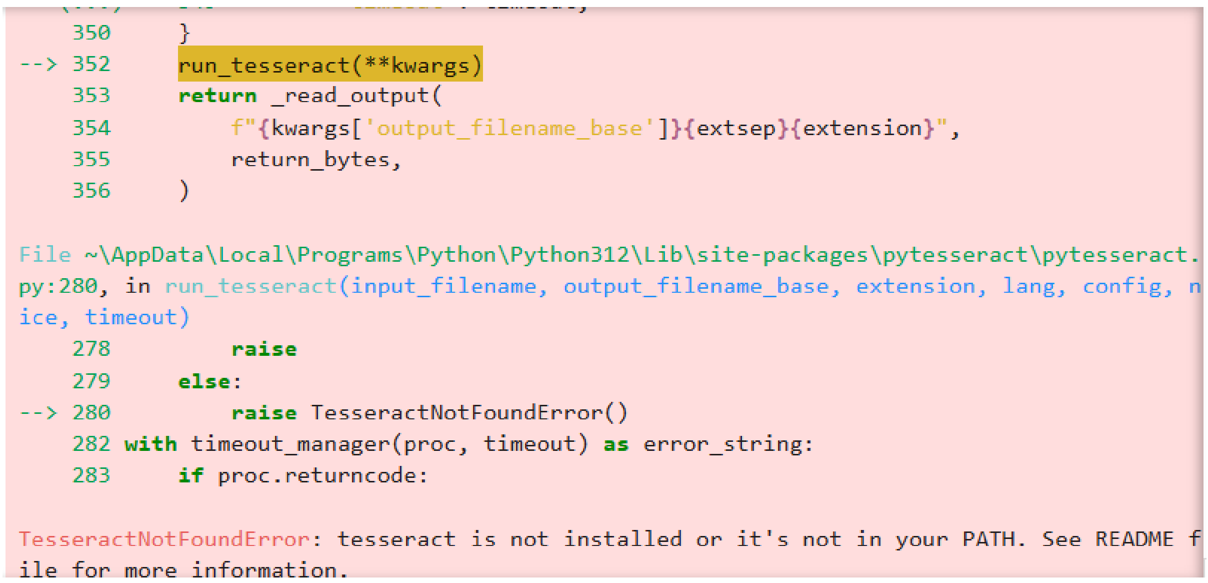
images = convert_from_path("sample_22mic0019.pdf", dpi=300, poppler_path=r"D:\7TH SEMESTER\Adv_Python\Python_lab\poppler\poppler-24.07.0\Library\bin")

What is Poppler in Python?

Poppler is a **PDF rendering library** originally developed for the Xpdf project. In the Python world, it's **not a Python package itself**, but rather a **set of command-line utilities** (like pdftoppm, pdfinfo, etc.) that help in rendering and processing PDF files.

Python libraries like pdf2image, pdfplumber, or PyMuPDF sometimes **rely on Poppler** tools under the hood to convert PDFs into images or extract information.

ERROR:

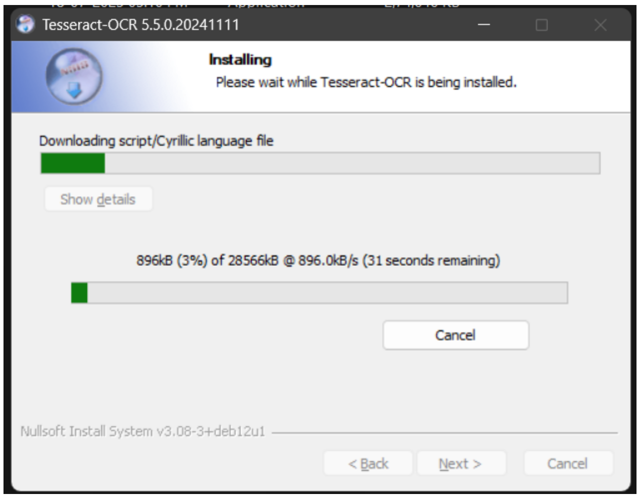


```
    350     }
--> 352     run_tesseract(**kwargs)
    353     return _read_output(
    354         f"{kwargs['output_filename_base']}{extsep}{extension}",
    355         return_bytes,
    356     )

File ~\AppData\Local\Programs\Python\Python312\Lib\site-packages\pytesseract\pytesseract.
py:280, in run_tesseract(input_filename, output_filename_base, extension, lang, config, n
ice, timeout)
    278         raise
    279     else:
--> 280         raise TesseractNotFoundError()
    282 with timeout_manager(proc, timeout) as error_string:
    283     if proc.returncode:

TesseractNotFoundError: tesseract is not installed or it's not in your PATH. See README f
ile for more information.
```

FIX:



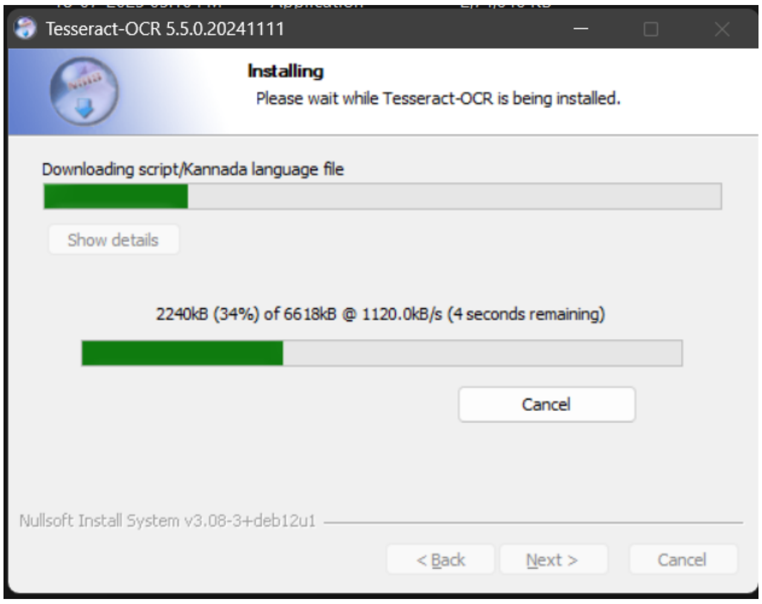
Tesseract-OCR 5.5.0.20241111

Installing

Please wait while Tesseract-OCR is being installed.

Downloading script/Cyrillic language file

896kB (3%) of 28566kB @ 896.0kB/s (31 seconds remaining)



Tesseract-OCR 5.5.0.20241111

Installing

Please wait while Tesseract-OCR is being installed.

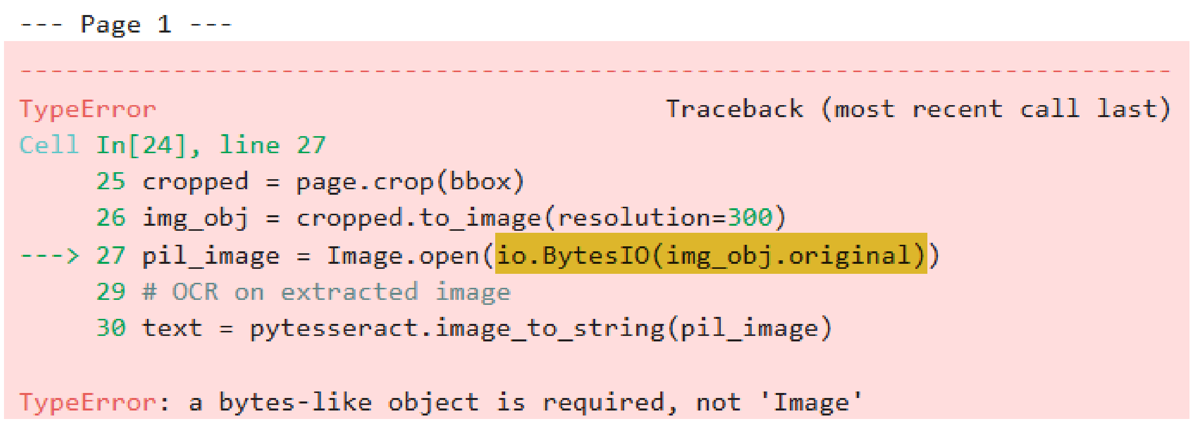
Downloading script/Kannada language file

2240kB (34%) of 6618kB @ 1120.0kB/s (4 seconds remaining)

Summary of Common OCR Issues that were observed from my output and their fixes

Type	**Example**	**Reason**	**Fix**
Spacing	emp loyee_df	Misjudged character spacing	Preprocessing, --psm 6
Substitution	I vs 1, O vs 0	Font/contrast confusion	High DPI, language model
Noise	WON AUBWNEH	Toolbar or UI misread	Crop only code area
Order	OUTPUT: at wrong place	Page layout misinterpretation	Layout-aware OCR
Smart quotes	“Judy”	Fancy fonts	Normalize post-OCR

Converting the **entire PDF page to image** when we **only need to extract text from an embedded image** is resource-intensive. So we try to detect the image in each page using bouneing boxes.



```
--- Page 1 ---
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[24], line 27
     25 cropped = page.crop(bbox)
     26 img_obj = cropped.to_image(resolution=300)
---> 27 pil_image = Image.open(io.BytesIO(img_obj.original))
     29 # OCR on extracted image
     30 text = pytesseract.image_to_string(pil_image)

TypeError: a bytes-like object is required, not 'Image'
```

TypeError: a bytes-like object is required, not 'Image'

happens because img_obj.original is already a **PIL Image object**, not raw image bytes. So you don't need to wrap it with Image.open(io.BytesIO(...)) — it's already in the correct format for Tesseract.

Fix

Just pass img_obj.original directly to pytesseract.image_to_string:

Python:

```
text = pytesseract.image_to_string(img_obj.original)
```