Internship Project

report on

# Extract and organize information in images with AI using IBM services.

**Submitted by:**

| | |
|---|---|
| Soundarya Ramesh Raykar | 4BD18CS098 |
| Sowmya N M | 4BD18CS099 |
| T Kavya | 4BD18CS115 |
| Reshmabanu M Badiger | 4BD19CS407 |

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**BAPUJI INSTITUTE OF ENGINEERING AND TECHNOLOGY**

**DAVANGERE**

ACADEMIC YEAR:2021-2022

# CONTENTS

| CHAPTERS | PAGE NO'S |
|---|---|
| 1. **INTRODUCTION** | 01 |
| 1.1 Overview | |
| 1.2 Purpose | |
| 2. **LITERATURE SURVEY** | 02-03 |
| 1.3 Existing Problem | |
| 1.4 Proposed Solution | |
| 3. **THEORETICAL ANALYSIS** | 04 |
| a. Hardware and Software Design | |
| 4. **EXPERIMENTAL INVESTIGATIONS** | 05-06 |
| a. Overview | |
| b. Architecture | |
| 5. **FLOWCHART** | 07 |
| 6. **RESULT** | 08-10 |
| 7. **ADVANTAGES AND DISADVANTAGES** | 11 |
| 8. **APPLICATIONS** | 12 |
| 9. **CONCLUSION** | 13 |
| 10. **FUTURE SCOPE** | 13 |
| 11. **BIBLIOGRAPHY** | 14-16 |

# 1 INTRODUCTION

## 1.1 Overview

In the digital era almost, everything is automated and information is stored and communicated in digital frame. However, there are several situations where the data is not digital and it might become essential to extract text from those to store in different point. The project aims at creating an application form where the user can upload a pdf document/Image containing text, the document is analyzed by an Optical character recognition (OCR) to extract text from it. The extracted text is again saved in a text document in the local drive. With the advent of OCR techniques, much time has been saved by automatically extracting the text out of a digital image of any invoice or a document. Currently, this is where most organizations that use OCR for any form of automation are Digital copies of invoices or documents are obtained by scanning or taking pictures. The text is extracted from these documents is entered into a template-based data entry software.

## 1.2 Purpose

- The project aims at creating an application form where the user can upload a pdf document/Image containing text.
- The document is analyzed by an Optical character recognition (OCR) to extract text from it.
- The extracted text is again saved in a text document in the local drive.

# 2 LITERATURE SURVEY

Hundreds of approaches have been proposed for text object extraction in image and pdf. In this chapter, we concentrate on the progress made after 2003 and review recently proposed text detection and localization approaches, text tracking approaches, and text extraction-based applications. Based on the literature review, we discuss the state of the art and the limitations of text extraction systems.

## 2.1 Existing Problem

### 2.1.1 Detection and Localization Approaches

In the text detection and localization approaches are divided into two categories: (i) Region based approaches that use the different properties between text region and background region to separate text objects. This category typically works in a bottom-up way by separating the image into small regions and then grouping small candidate regions into text regions; (ii) Texture based approaches that use distinct texture properties of text to extract text from background. This category typically works in a top-down way by extracting texture features of image and then locating text regions, we follow this categorization and further subdivide the region-based approaches based on the features used in the approaches.

### 2.1.2 Text Tracking Approaches

Text tracking stage utilizes the temporal redundancy and spatial consistency properties of text objects in video documents to speedup detection and localization of the same text in consecutive frames. Moreover, Text tracking also serves as a text verification stage to remove false alarms. Compared to text detection and localization, however, only a few works are reported for text tracking in recent years.

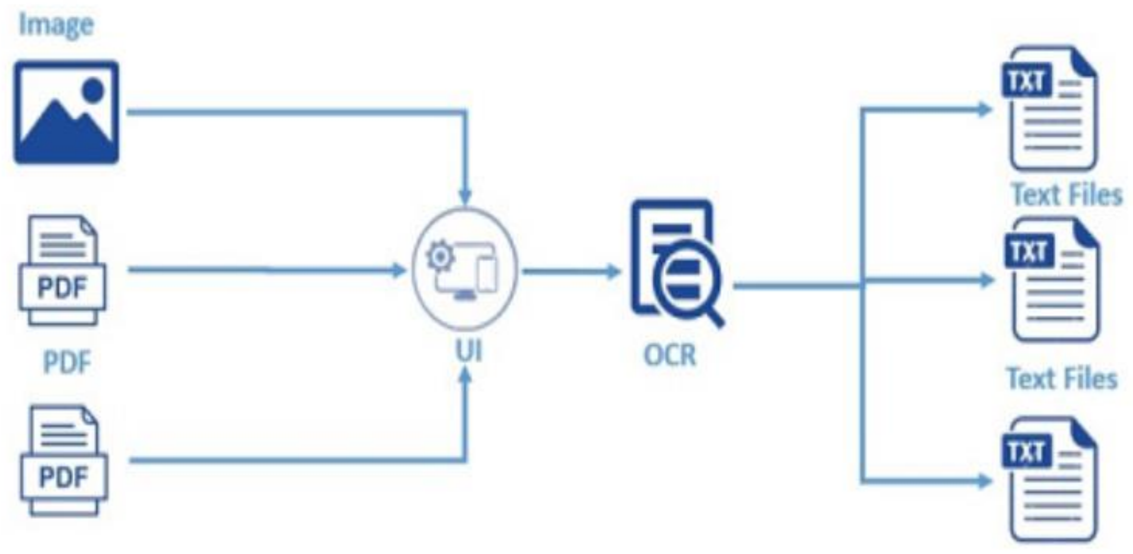
## 2.2 Proposed Solution

In the proposed text model, each character is a part and every two neighboring parts are connected by a link. For every part in the model, we use the presented character features to compute character energy, which can reflect the inherent properties of characters and indicate the probability that a candidate part in the model is a character. For every link in the model, we use the spatial relationship and property similarity between neighboring characters to compute link energy, which indicates the probability that two connected candidate parts are both characters.

The advantages of the proposed method are: (i) the characteristics of character and the structure of text object are described by the parts and links of the presented text model. Therefore, the proposed method can capture the properties of characters and text objects simultaneously and combine them efficiently; (ii) the presented three new character features are computed based on the inherent properties of character. Therefore, the proposed method is robust to the size, font, color, and orientation of text and can discriminate text objects from other objects efficiently.

# 3 THEORITICAL ANALYSIS

## 3.1 Block Diagram



## 3.2 Hardware and Software Designing

Hardware Requirements:

| | |
|---|---|
| Processor | : Intel Core i3 |
| Processor Speed | : 1.7 GHz |
| RAM | : 4 GB |
| System Type | : 64-Bit Operating System |
| Monitor Resolution | : 1280*800 |

Software Requirements:

| | |
|---|---|
| Software | : Spyder ( Anaconda3 ) |
| Operating System | : Windows 10 |
| Front End | : HTML, CSS |
| Programming Language | : Python |

# 4 EXPERIMENTAL INVESTIGATIONS

## Tesseract:

Tesseract is an open-source text recognition engine that is available under the Apache 2.0 license and its development has been sponsored by Google since 2006. In the year 2006, Tesseract was considered as one of the most accurate open-source OCR engines. We can use it directly or can use the API to extract the printed text from images. The best part is that it supports an extensive variety of languages. It is through wrappers that Tesseract can be made compatible with different programming languages and frameworks. Python wrapper named tesseract, is used to recognize text from a large document, or it can also be used to recognize text from an image of a single text line.

## OpenCV:

OpenCV (Open Source Computer Vision Library), as the name suggests, is an open source computer vision and a machine learning software library. OpenCV was built to furnish a common infrastructure for computer vision applications. Besides this, it also accelerates the use of machine perception in commercial products. And being a BSD-licensed product, OpenCV comes in handy for businesses to utilize and modify the code. The library comprises more than 2500 optimized algorithms that have a comprehensive set of both classic and modern computer vision along with the machine learning algorithms. These algorithms can be used in detecting and recognizing faces, in classifying human actions in videos, extracting 3D models of objects, and many more. Tesseract requires a clean image to detect the text, this is where OpenCV plays an important role as it performs the operations on an image like converting a colored image to binary image, adjusting the contrast of an image, edge detection, and many more.

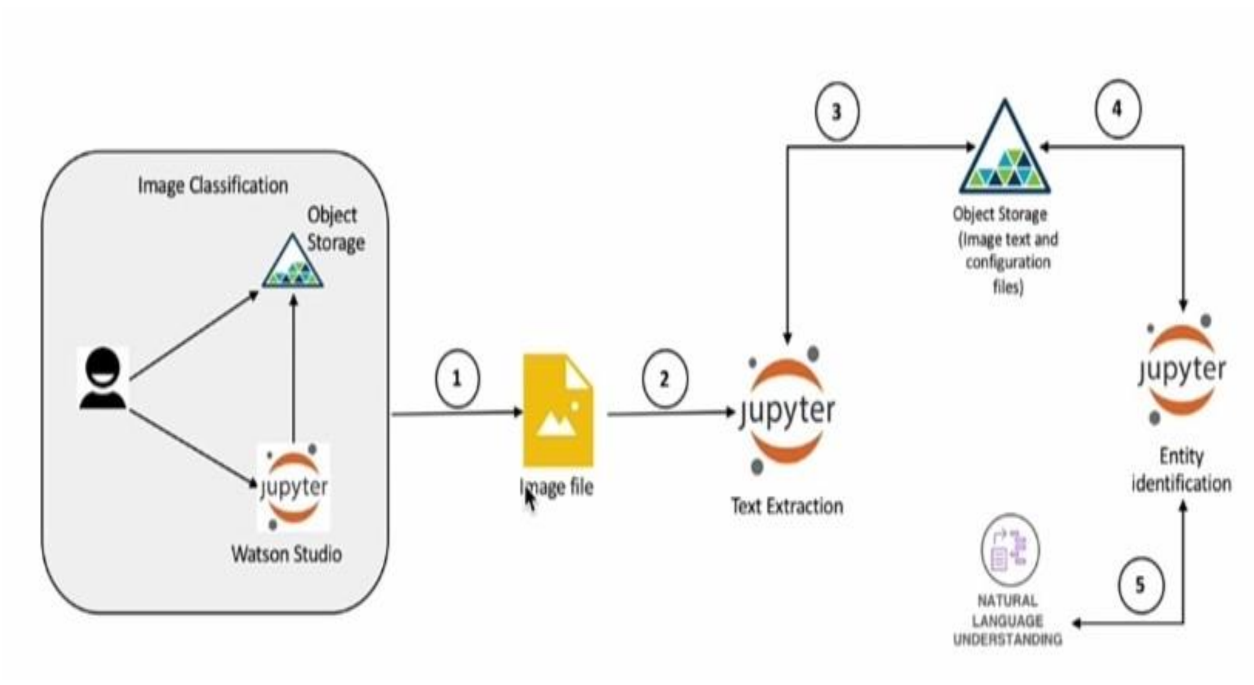
## Poppler:

Plopper is a free software utility library for rendering Portable Document Format (PDF) documents. Its development is supported by freedesktop.org. It is commonly used on Linux systems,[3] and is used by the PDF viewers of the open source GNOME and KDE desktop environments. Poppler can use two back-ends for drawing PDF documents, Cairo and Splash. Its features may depend on which back-end it employs. A third back-end based on Qt4's painting framework "Arthur", is available, but is incomplete and no longer under active development. Bindings exist for Glib and Qt5, that provide interfaces to the Poppler back ends, although the Qt5 bindings support only the Splash and Arthur back ends. There is a patch set available to add support for the Cairo backend to the Qt5 bindings, but the Poppler project does not currently wish to integrate the feature into the library proper.

## Python flask:

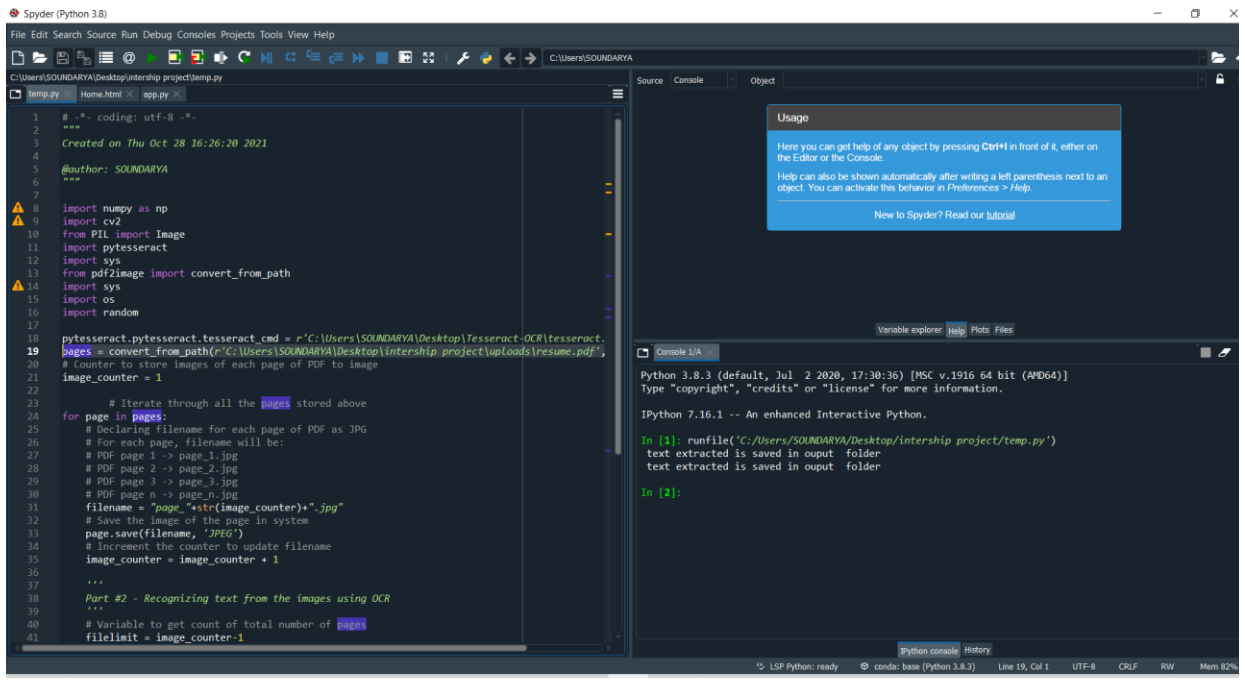
Flask is a micro web framework written in Python. Extensions exists for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Flask is used for the backend, but it makes use of a templating language called Jinja2 which is used to create HTML, XML or other markup formats that are returned to the user via an HTTP request. Flask offers a diversified working style while Django offers a Monolithic working style.It is designed as a web framework for restful API development.

## 5 FLOWCHART



The Figure shows the extraction of text in the image and the pdf file.

# 6 RESULT



```python
# -*- coding: utf-8 -*-
"""
Created on Thu Oct 28 16:26:20 2021

@author: SOUNDARYA
"""

import numpy as np
import cv2
from PIL import Image
import pytesseract
import sys
from pdf2image import convert_from_path
import sys
import os
import random

pytesseract.pytesseract.tesseract_cmd = r'C:\Users\SOUNDARYA\Desktop\Tesseract-OCR\tesseract.
pages = convert_from_path(r'C:\Users\SOUNDARYA\Desktop\intership project\uploads\resume.pdf',
# Counter to store images of each page of PDF to image
image_counter = 1

        # Iterate through all the pages stored above
for page in pages:
    # Declaring filename for each page of PDF as JPG
    # For each page, filename will be:
    # PDF page 1 -> page_1.jpg
    # PDF page 2 -> page_2.jpg
    # PDF page 3 -> page_3.jpg
    # PDF page n -> page_n.jpg
    filename = "page_"+str(image_counter)+".jpg"
    # Save the image of the page in system
    page.save(filename, 'JPEG')
    # Increment the counter to update filename
    image_counter = image_counter + 1

    '''
    Part #2 - Recognizing text from the images using OCR
    '''
    # Variable to get count of total number of pages
    filelimit = image_counter-1
```

Usage

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

New to Spyder? Read our tutorial

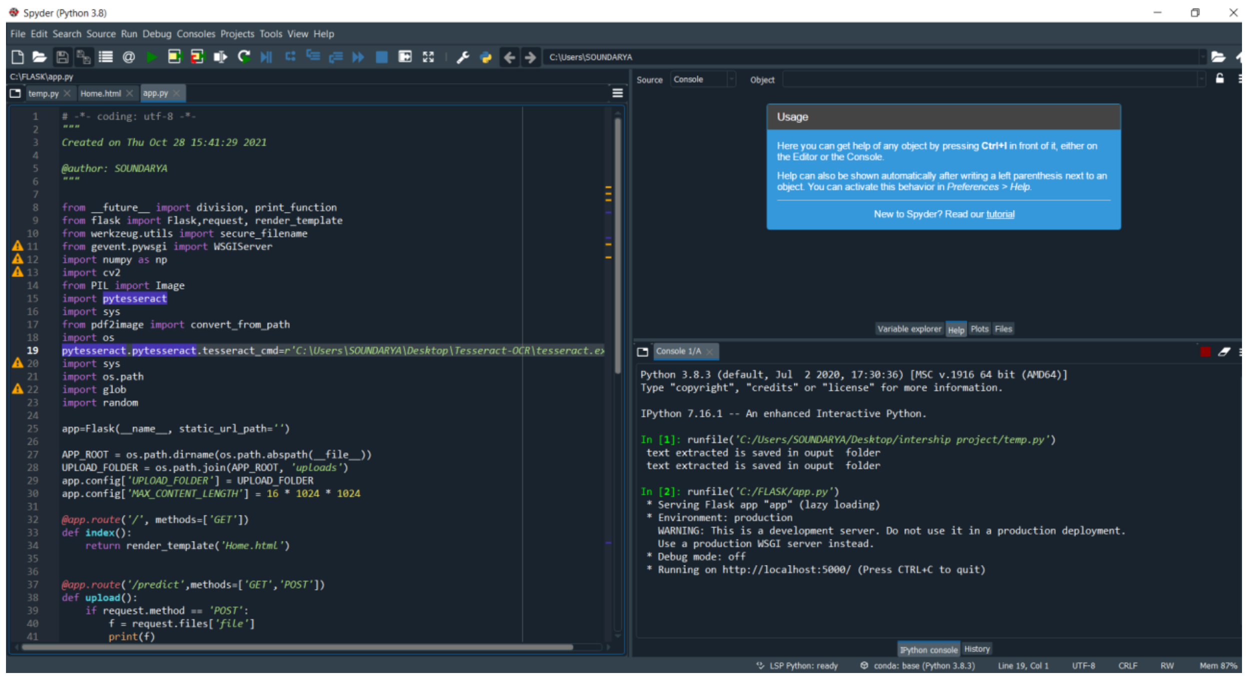
```
Python 3.8.3 (default, Jul  2 2020, 17:30:36) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.16.1 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/SOUNDARYA/Desktop/intership project/temp.py')
 text extracted is saved in ouput  folder
 text extracted is saved in ouput  folder

In [2]:
```

The Fig 6.1 displays Output of the Text Extraction Code.



```python
# -*- coding: utf-8 -*-
"""
Created on Thu Oct 28 15:41:29 2021

@author: SOUNDARYA
"""

from __future__ import division, print_function
from flask import Flask,request, render_template
from werkzeug.utils import secure_filename
from gevent.pywsgi import WSGIServer
import numpy as np
import cv2
from PIL import Image
import pytesseract
import sys
from pdf2image import convert_from_path
import os
pytesseract.pytesseract.tesseract_cmd=r'C:\Users\SOUNDARYA\Desktop\Tesseract-OCR\tesseract.ex
import sys
import os.path
import glob
import random

app=Flask(__name__, static_url_path='')

APP_ROOT = os.path.dirname(os.path.abspath(__file__))
UPLOAD_FOLDER = os.path.join(APP_ROOT, 'uploads')
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
app.config['MAX_CONTENT_LENGTH'] = 16 * 1024 * 1024

@app.route('/', methods=['GET'])
def index():
    return render_template('Home.html')


@app.route('/predict',methods=['GET','POST'])
def upload():
    if request.method == 'POST':
        f = request.files['file']
        print(f)
```

Usage

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

New to Spyder? Read our tutorial

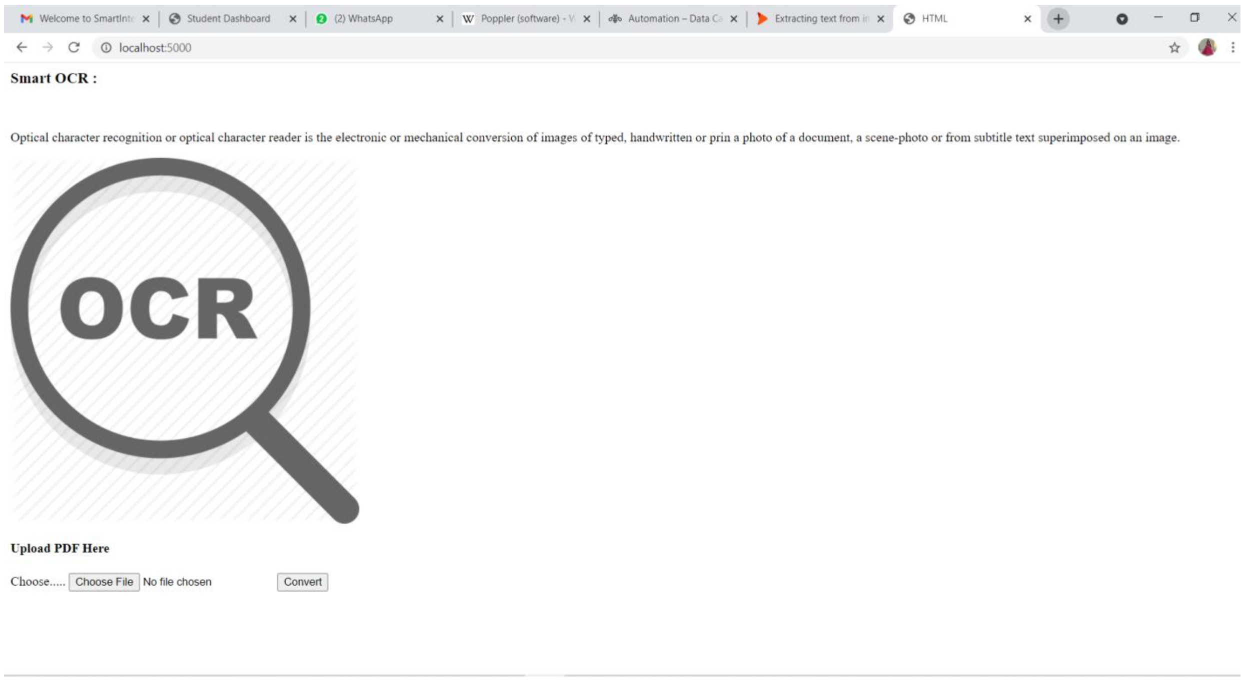
```
Python 3.8.3 (default, Jul  2 2020, 17:30:36) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.16.1 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/SOUNDARYA/Desktop/intership project/temp.py')
 text extracted is saved in ouput  folder
 text extracted is saved in ouput  folder

In [2]: runfile('C:/FLASK/app.py')
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://localhost:5000/ (Press CTRL+C to quit)
```

The Fig 6.2 displays Output of the Python Code



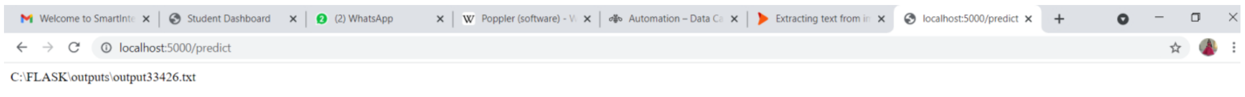
**Smart OCR :**

Optical character recognition or optical character reader is the electronic or mechanical conversion of images of typed, handwritten or prin a photo of a document, a scene-photo or from subtitle text superimposed on an image.

**Upload PDF Here**

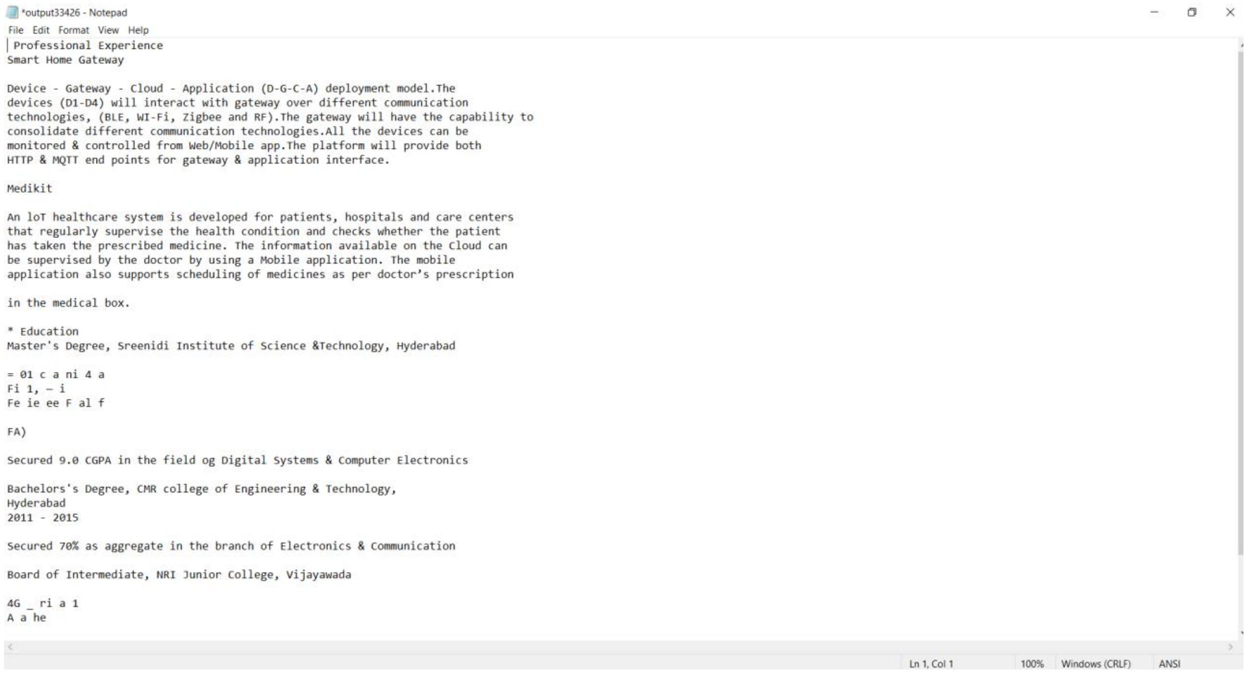
Choose..... Choose File No file chosen Convert

The Fig 6.3 displays Smart OCR.



```
C:\FLASK\outputs\output33426.txt
```

The Fig 6.4 shows the location of the output folder.



```
Professional Experience
Smart Home Gateway

Device - Gateway - Cloud - Application (D-G-C-A) deployment model.The
devices (D1-D4) will interact with gateway over different communication
technologies, (BLE, WI-Fi, Zigbee and RF).The gateway will have the capability to
consolidate different communication technologies.All the devices can be
monitored & controlled from Web/Mobile app.The platform will provide both
HTTP & MQTT end points for gateway & application interface.

Medikit

An loT healthcare system is developed for patients, hospitals and care centers
that regularly supervise the health condition and checks whether the patient
has taken the prescribed medicine. The information available on the Cloud can
be supervised by the doctor by using a Mobile application. The mobile
application also supports scheduling of medicines as per doctor's prescription

in the medical box.

* Education
Master's Degree, Sreenidi Institute of Science &Technology, Hyderabad

= 01 c a ni 4 a
Fi 1, — i
Fe ie ee F al f

FA)

Secured 9.0 CGPA in the field og Digital Systems & Computer Electronics

Bachelors's Degree, CMR college of Engineering & Technology,
Hyderabad
2011 - 2015

Secured 70% as aggregate in the branch of Electronics & Communication

Board of Intermediate, NRI Junior College, Vijayawada

4G _ ri a 1
A a he
```

The Fig 6.4 displays the text document where the output is stored.

# 7 ADVANTAGES AND DISADVANTAGES

## Advantages:

- Tilt text is detected
- High accuracy in natural scene
- Requires less text extraction database
- Most relevant and accurate data is retrieved from the web

## Disadvantages:

- Handwritten text can't be accurately recognized.

# 8 APPLICATIONS

Clearly, the most important and widely used application of a text extraction system is text-based image/video indexing and retrieval by using the recognized outputs of the system. Besides that, many other applications of text extraction system have been developed as well, such as camera-based text reading system for visually impaired people, wearable translation robot, wearable text-tracking system, vehicle license plate recognition, and road sign text detection for mobile device.

# 9 CONCLUSION

Text objects occurring in image and video documents can provide much useful information for content based information retrieval and indexing applications, because they contain much semantic information related to the documents' contents. However, extracting text from images and videos is a very difficult task due to the varying font, size, color, orientation, and deformation of text objects. Although a large number of text extraction approaches have been reported in the literature, no specific designed text model and character features are presented to capture the unique properties and structure of characters and text objects

# 10 FUTURE WORK

Future investigations on other aspects need to be pursued for developing solid image to text detection and recognition applications and related multimedia retrieval and annotation applications.

## 11 BIBILOGRAPHY

- https://github.com/IBM/image-recognition-and-information-extraction-from-image-documents.
- https://www.toolbox.com/hr/hr-analytics/articles/what-is-hr-analytics/
- https://www.ibm.com/downloads/cas/WVJBQERV.
- https://www.ibm.com/docs/en/SSEP7J_11.1.0/com.ibm.swg.ba.cognos.ug_ca_dshb.doc/ug_ca_dshb.pdf.
- https://en.wikipedia.org/wiki/IBM_Cognos_Analytics.

## Appendix:

**Source Code**

```
from __future__ import division, print_function

from flask import Flask,request, render_template

from werkzeug.utils import secure_filename

from gevent.pywsgi import WSGIServer

import numpy as np

import cv2

from PIL import Image

import pytesseract

import sys

from pdf2image import convert_from_path

import os

pytesseract.pytesseract.tesseract_cmd=r'C:\Users\SOUNDARYA\Desktop\Tesseract-
OCR\tesseract.exe'
```

```
import sys

import os.path

import glob

import random

app=Flask(__name__, static_url_path='')

APP_ROOT = os.path.dirname(os.path.abspath(__file__))

UPLOAD_FOLDER = os.path.join(APP_ROOT, 'uploads')

app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

app.config['MAX_CONTENT_LENGTH'] = 16 * 1024 * 1024


@app.route('/', methods=['GET'])

def index():

    return render_template('Home.html')

@app.route('/predict',methods=['GET','POST'])

def upload():

    if request.method == 'POST':

        f = request.files['file']

        print(f)

        basepath= os.path.dirname(__file__)

        print(basepath)

       file_path=os.path.join(app.config['UPLOAD_FOLDER'],secure_filename(f.filename))

        print(file_path)

        print(secure_filename(f.filename))
        f.save(file_path)
        PDF_file=file_path
        pages=convert_from_path(PDF_file,500)
        image_counter=1
        for page in pages:
            filename="page_"+str(image_counter)+".jpg"
            page.save(filename,'JPEG')
        image_counter=image_counter+1
        filelimit=image_counter-1
        basepath=os.path.dirname(__file__)
        file_path2=os.path.join(basepath,'outputs',"output"+str(random.randint(1,100000))+".txt")
        f=open(file_path2,"a")
        for i in range(1,filelimit+1):
            filename="page_"+str(i)+".jpg"
            text=str(((pytesseract.image_to_string(Image.open(filename)))))
            text=text.replace('-\n','')
            f.write(text)
            f.close()
            return file_path2

if __name__ == '__main__':
    app.run(host="localhost", debug=False)
```