CHAPTER-1
INTRODUCTION

Traditionally, data enters the system through the use of a keyboard. The main disadvantage of such an arrangement is the reliance on data input by the user (i.e. the human being). While the keyboard is still considered a revolution in the digital era, it may not always be the optimal option for personal data entry or movement. Data entry using keyboards is often time-consuming, unreliable, and inefficient. As a result, the demand for other, mostly faster ways of data identification increased1. These automatic identification methods mostly had different applications, e.g. transcription software, barcode scanners, QR-codes and optical character recognition.

Firms have become increasingly digitized over the past decades. Physical locations are rapidly closing, while doing business and tasks over the internet has increased significantly over the years. This gave rise to serious new challenges in proving someone's identity. While someone can prove their identity by entering a physical location and showing their identity document, this is not possible over the internet. Furthermore, someone's identity on the internet is not always certain. Fraud and identity theft is not uncommon in the digital world. In india 45 percent of adult indian internet users faced threat in 2020,up almost 40 percent since 2019 to count at 2.7 crore over 2 percent of india's entire population. This raises questions on whether you can trust information provided by the user without verifying the user first. There is, therefore, a strong need for a robust digital identity infrastructure that takes into account the current limitations. As privacy awareness has increased significantly throughout the years and will continue to increase in the coming years.

At DID (Digitization of Indian Documents using OCR) system, we are aware of the current challenges of digital identity verification. We developed an identity verification service platform that verifies identity documents using the camera of the user. What separates DID system from other parties, is its privacy- first approach: data processing is only performed on the device of the user and is not stored on our servers.

This paper mainly focuses on the use of optical character recognition systems and its application by DID system. The main topics that are discussed are the implementation of OCR at DID system, the challenges of identity verification using OCR, how DID system addresses these challenges

1.1 STATEMENT OF THE PROBLEM

Digital identity is the cornerstone of a complete digital experience that customers now expect. Common tasks such as enrolling for a new service, logging into an existing service, making changes to an existing account, or performing a payment all rely on customers being able to prove who they are. Without a verified digital identity, companies open themselves up to fraud, regulatory penalties and exposure to other security risks. It is with this in mind we are on this venture to develop a web based application for verifying the identity. We propose DID for detecting his event. After event detection, we extract the data we want from the document and store it for future use. The data extracted from the image is cross matched to prove whether it is valid or forged document.some of the information from the document is uniquely used to identify and also ensure that the cross checked data is same in both.

CHAPTER-2
SYSTEM ANALYSIS

2.1 PRESENT SYSTEM

Traditional technique available for different service providers like government organisations , telecom and other sectors used are out dated methods and anyone can make fraud. The technique that used traditionally for verifying the identity is through document verification which is signed by a designated person by cross checking different other related documents and it is a time taking process. And also anyone can make forged documents and submit at verification process. Thus the traditionally system is inefficient and non reliable.

2.2 LIMITATIONS OF PRESENT SYSTEM

a) Documents submitted for verification can be forged one
b) The process of verification is time taking process.
c) Reduces operational efficiency
d) Authorities who are verifying can be bribe by personal

2.3 PROPOSED SYSTEM

DID system is proposed for extracting the data that we want from documents such as pan card, aadhar card, driving licence, voter id card and cross matched every information that extracted from the document to identify where any one of the data in the scanned image is valid data or to be a forged one. The only requirement of this system is installing special cameras for capturing the documents.The quality of image that captured is most important in this proposed system then only it can provide more accuracy.

2.4 ADVANTAGES AND FEATURES OF PROPOSED SYSTEM

Various Service providers like government, telecom and other sectors can leverage identity verification technology to

a) Reduce identity fraud
b) Adapt to local regulations and business needs
c) Streamline customer on boarding across multiple channels
d) comply with know your customer (KYC) regulations : Anti Money laundering, Combating the financing terrorism

Features of proposed system

SI.NO	FEATURES
1	REMOTE ID CAPTURE
2	EXTRACTING INFORMATION
3	CROSS MATCHING DOCUMENT
4	VERIFICATION RESPONSE

1.Remote ID Capture

Document capture is done by providing real-time indications to user to help them to scan their ID with device camera .The device camera enables to capture the document data by taking a picture or simply uploading the documents.

2. Extracting Information

From captured image the data is extracted through optical character recognition.Optical Character Recognition is a technique of reading or grabbing text from scanned photos, handwritten images and convert it into a digital format that can be editable and searchable. In our system the extracted information from different documents contains unwanted datas from this data we cleanup those junk data and make the data that we want more clarity.

Here we use Tesseract for extracting informations from an image.This enable the DID system to accurately decipher and extract text from a variety of sources .As per its's namesake it uses an updated version of tesseract open source ocr tool. We also automatically binarize and preprocess images using the binarization so tesseract has an easier time deciphering images

3. Cross Matching Document

The data extracted from the document is then cross verified to ensure the authenticity of th person .For cross matching it checked some important data from each of the document.considering the paancard as an example it cross checks mainly for date of birth and pan number .why we cross checking these data is that in most of cases the date of birth and pan no is forged .In different scenarios we cross checked different datas

4. Verification Response

DID system cross checked the important data with coressponding api and return the matching document. In DID system for different scenario it cross checks different other data for each of the document in this system and checks in api with a particular id such as aadhar number, pan number, licence number, voter id number and from each of the response it cross checks again the information in those id contain data for more validating the result

2.5 Feasibility Study

The main aim of the feasibility study activity is to determine. Whether it would be financially and technically feasible to develop the product. The feasibility study activity involves analysis of the problem and collection of all relevant information relating to the product such as the different data items which would be input to the system the processing required to be carried out of these data, the output data required to be carried out of these data, the output data required to be produced by the system, as well as various constraints on the behaviour of the system.

a) Technical Feasibility

b) Economic Feasibility

c) Operational Feasibility

2.5.1 Technical Feasibility

The proposed system uses language python.Based n this criteria we can strongly say that it is technically feasible ,since there will not be much difficulty n getting required resources for the development and maintain system as well. All the resources

needed for the development of the software as well as the maintenance of the same is available in the organisation. Here we are utilizing the resources which are already available so it is technically feasible that we can implement the DID system

2.5.2 Operational Feasibility

Proposed system would be beneficial only if they can be turned in to information system that will meet the organization operating requirements. One of the main problems faced during the development of a new system is getting acceptance from user.

2.5.3 Economic Feasibility

Economically, this project doesn't raise any issue, as the project itself is planned as the website. The resource required for this project is minimum. This system doesn't demand any additional equipment.

a) The resource required for our project is minimum and the system doesn't demand any additional equipment so our project is economically feasible

b) The system is economic with our system point of view

c) It is cost effective as it eliminates the paper works

CHAPTER – 3

SYSTEM SPECIFICATION

3.1 MINIMUM SOFTWARE REQUIREMENTS FOR DEVELOPMENT

1.Operating System	:	Windows / Mac Os
2.Language	:	Python
3.Frontend	:	HTML,CSS
4.Framework	:	Django:
5.IDE	:	Pycharm
6.Libraries	:	Opencv, Pytesseract,

3.2 MINIMUM HARDWARE REQUIREMENTS FOR DEVELOPMENT

1.Processor	:	Intel i5 or AMD Ryzen 3
2.RAM	:	3 GB
3.Hard Disk Drive	:	300GB
4.Peripherals	:	Keyboard, Mouse, Camera, Monitor

CHAPTER – 4

SYSTEM DESIGN

4.1 CONTEXT LEVEL DIAGRAM

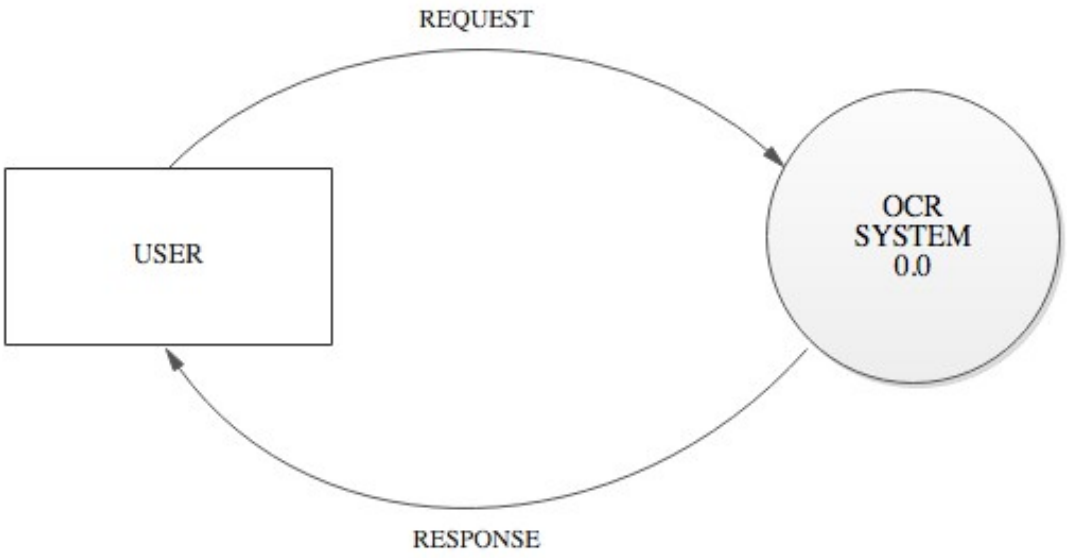


FIG 4.1.1 CONTEXT LEVEL DIAGRAM

4.2 DATA FLOW DIAGRAM

LEVEL 1: DFD OF DID SYSTEM

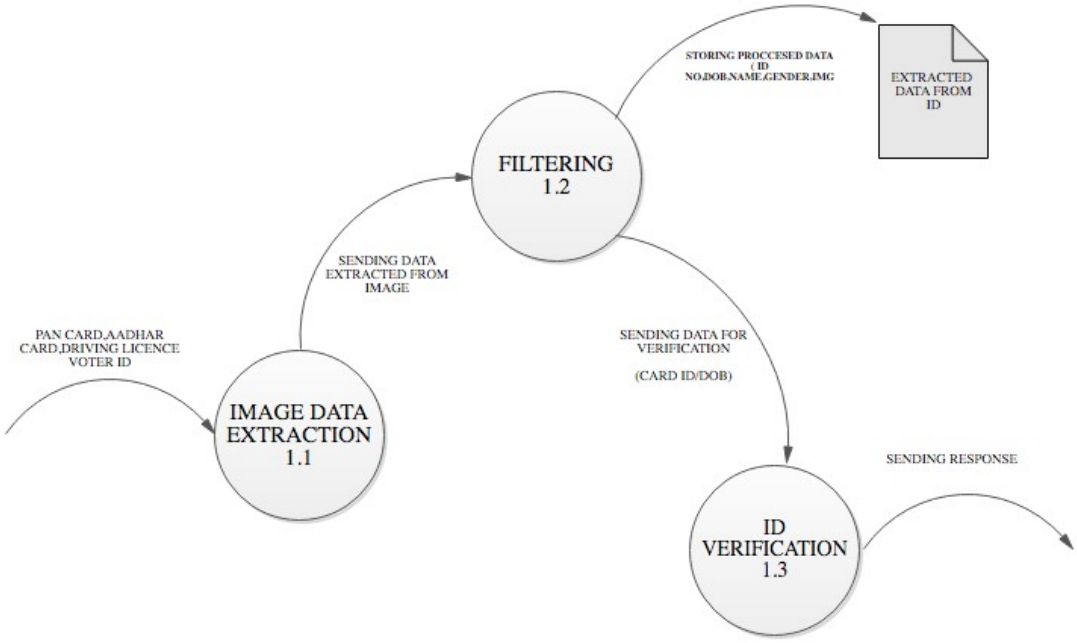


FIG 4.2.1 LEVEL 1 DFD: DID SYSTEM

4.3 DATABASE DIAGRAM

4.3.1 TABLE: PAN-CARD DATABASE

FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
Pan Number	CharField	100	PrimaryKey
Father Name	Char Field	100	Not Null
Name	Char Field	100	Not Null
Pan Dob	Char Field	100	Not Null
Pan Image	Image Field		Not Null

4.3.2 TABLE: AADHAR CARD DATABASE

FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
Aadhar Number	Char Field	100	PrimaryKey
Aadhar Name	Char Field	100	Not Null
Aadhar Birth	Char Field	100	Not Null
Aadhar Gender	Char Field	100	Not Null
Aadhar Image	Image Field		Not Null

4.3.4 TABLE: DRIVER LICENCE DATABASE

FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
Driver Licence No	Char Field	100	PrimaryKey
Driver Licence Name	Char Field	100	Not Null
Driver Licence L name	Char Field	100	Not Null
Driver Licence Image	Image Field		Not Null

4.3..5 TABLE: VOTER ID DATABASE

FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
Voter Father Name	Char Field	100	PrimaryKey
Voter Name	Char Field	100	Not Null
Voter Image	Image Field		Not Null

4.4 NORMALIZATION

Designing a data base is a complex task and the normalization theory is a useful and in this design process.

A bad database design may lead to certain undesirable situations such as;

a) Repetition of information
b) Inability to represent certain information
c) Loss of information

To minimize the anomalies, normalization may be used. In construction and system,

First Normal form (1 NF)

A relation is in first normal form (1 NF), if and only if all its attributes are based on a single domain. The objective of normalizing a table is to remove its repeating groups and ensure that all entries of the resulting table have at most single value. The objective of 1NF is to divide the database into logical units called tables. When each table has been designed, primary key is assigned to most or all tables.

Second Normal form (2 NF)

A table is said to be in second normal form (2 NF), when it is in 1NF and it satisfies functional dependency. Functional of 2NF is to take data that is partially dependent on the primary key, enter the data into another table. Now consider the data base of the system. In this there are a total of table and all tables are in second normal form. That is all of these tables satisfies second normal form. No repeated information is stored in any table. All of the table. All of the tables have a separate primary key some of all are auto numbers.

Third Normal form (3 NF)

A table is said to be in 3NF, when it is 2NF and every non-key attribute is functionally dependent only on the primary key. The objective of 3NF is to remove data in a table that is not dependent on the primary key.

All our tables are in First normalization form

4.5 DESIGN OF EACH SUB SYSTEM

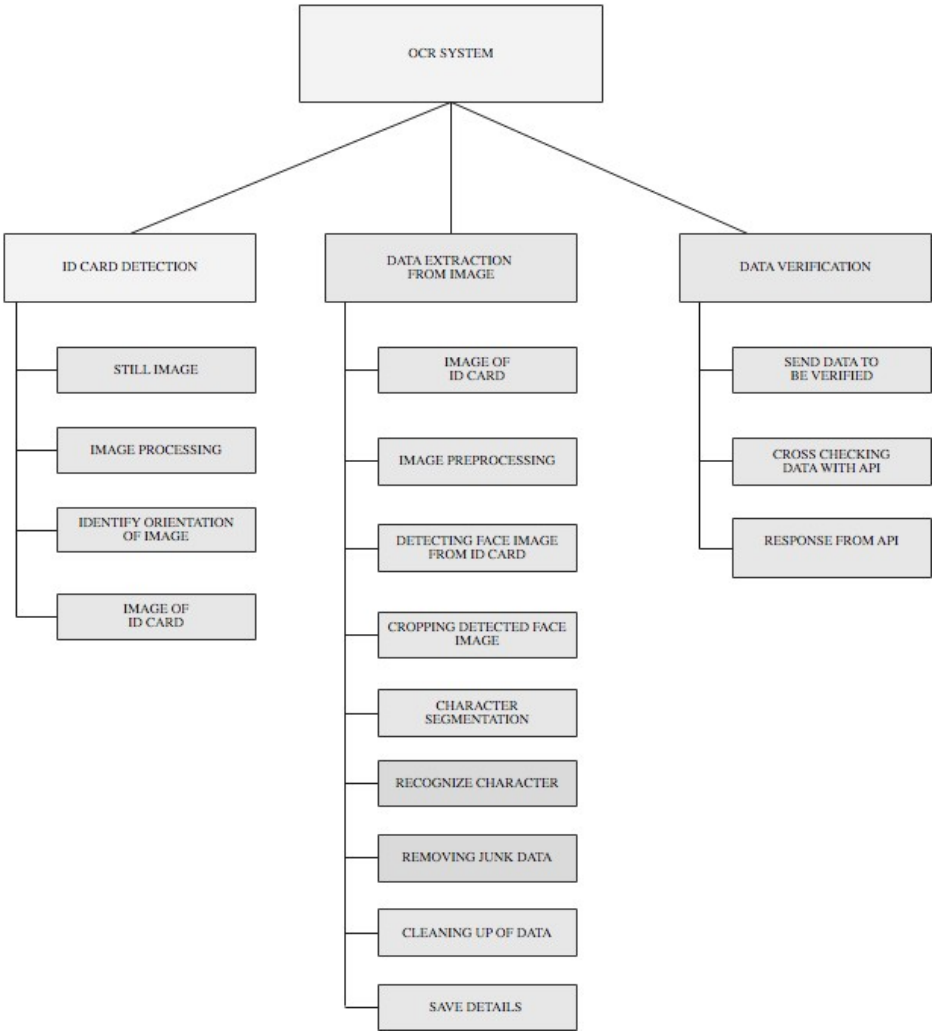


FIG 4.5.1 DESIGN OF SUB SYTEM

4.6 UML DIAGRAMS

4.6.1 USE CASE DIAGRAM

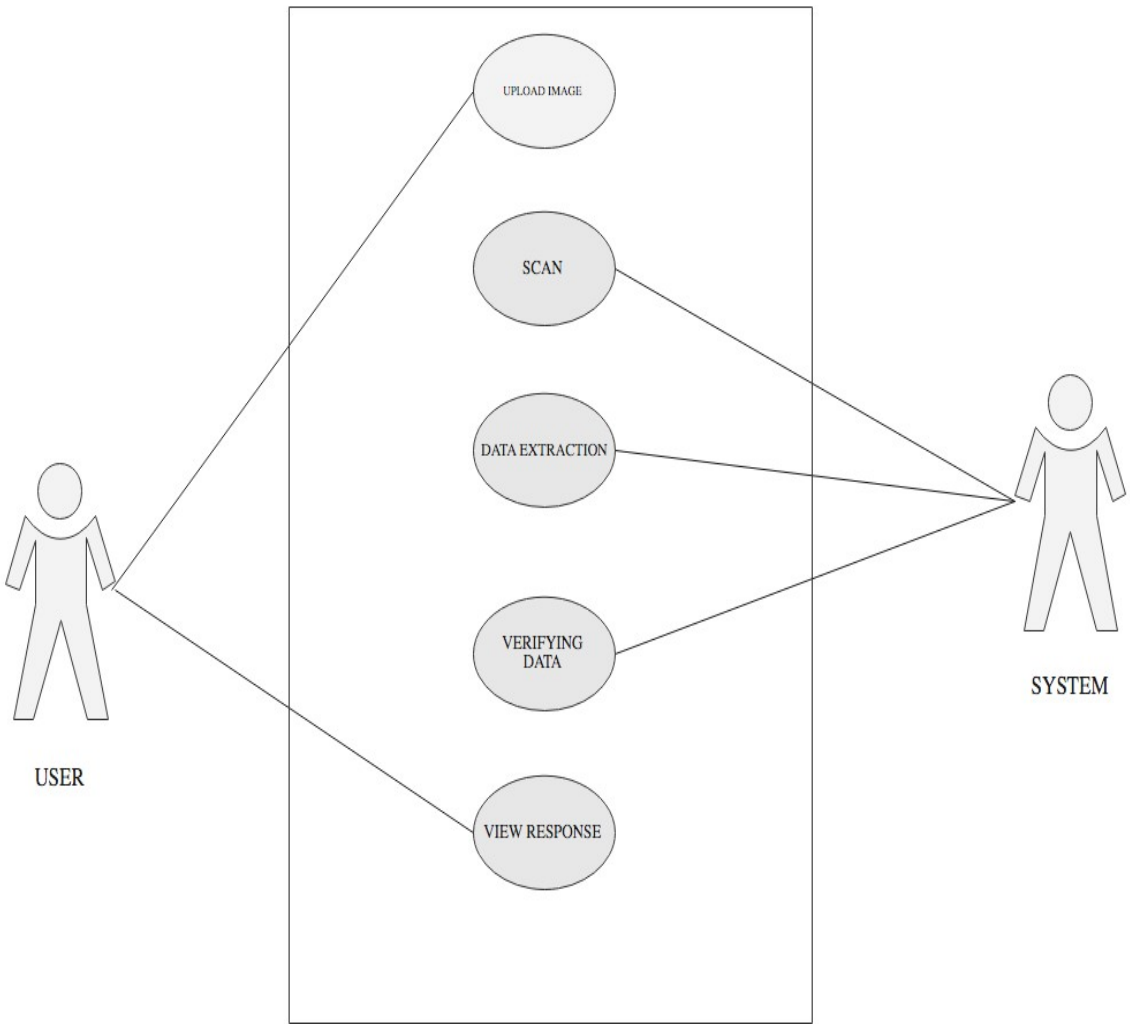


FIG 4.6.1 USE CASE DIAGRAM

4.6.2 SEQUENCE DIAGRAM

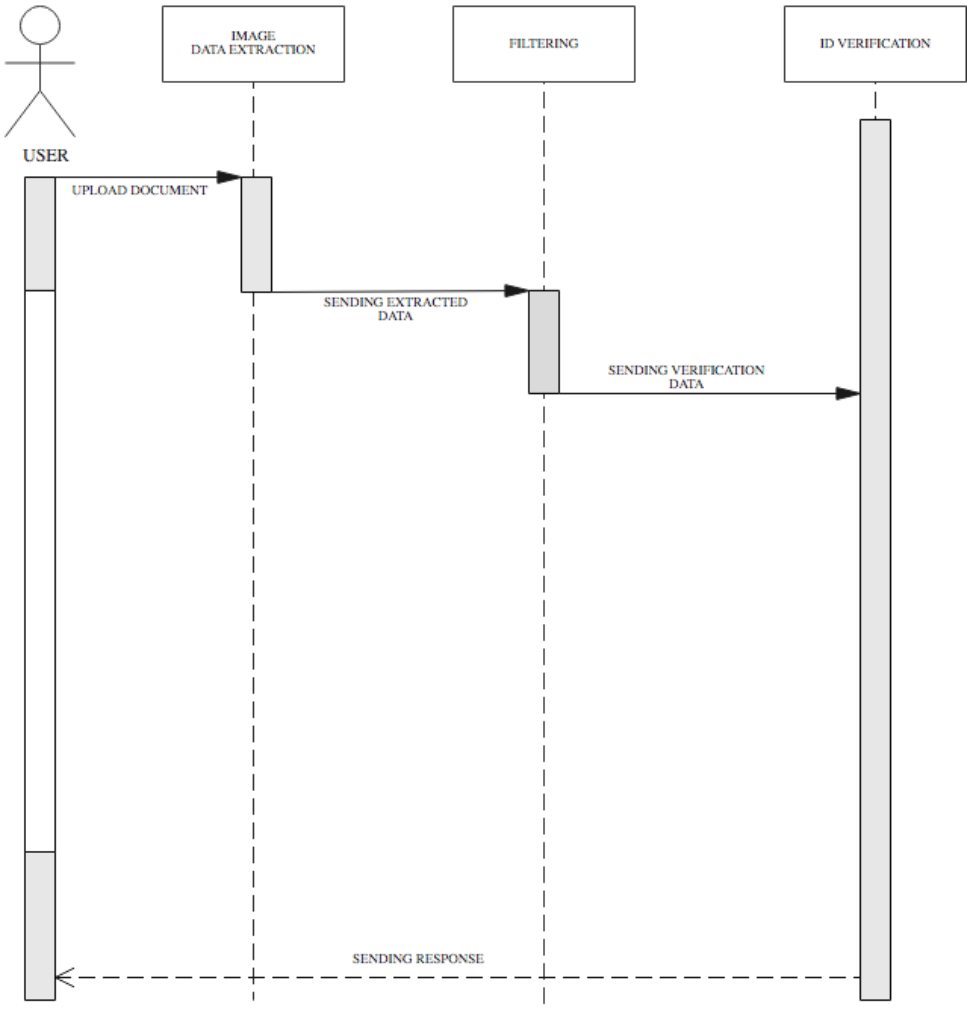


FIG 4.6.2 SEQUENCE DIAGRAM

4.7 FLOW CHART

4.7.1 PANCARD FLOWCHART

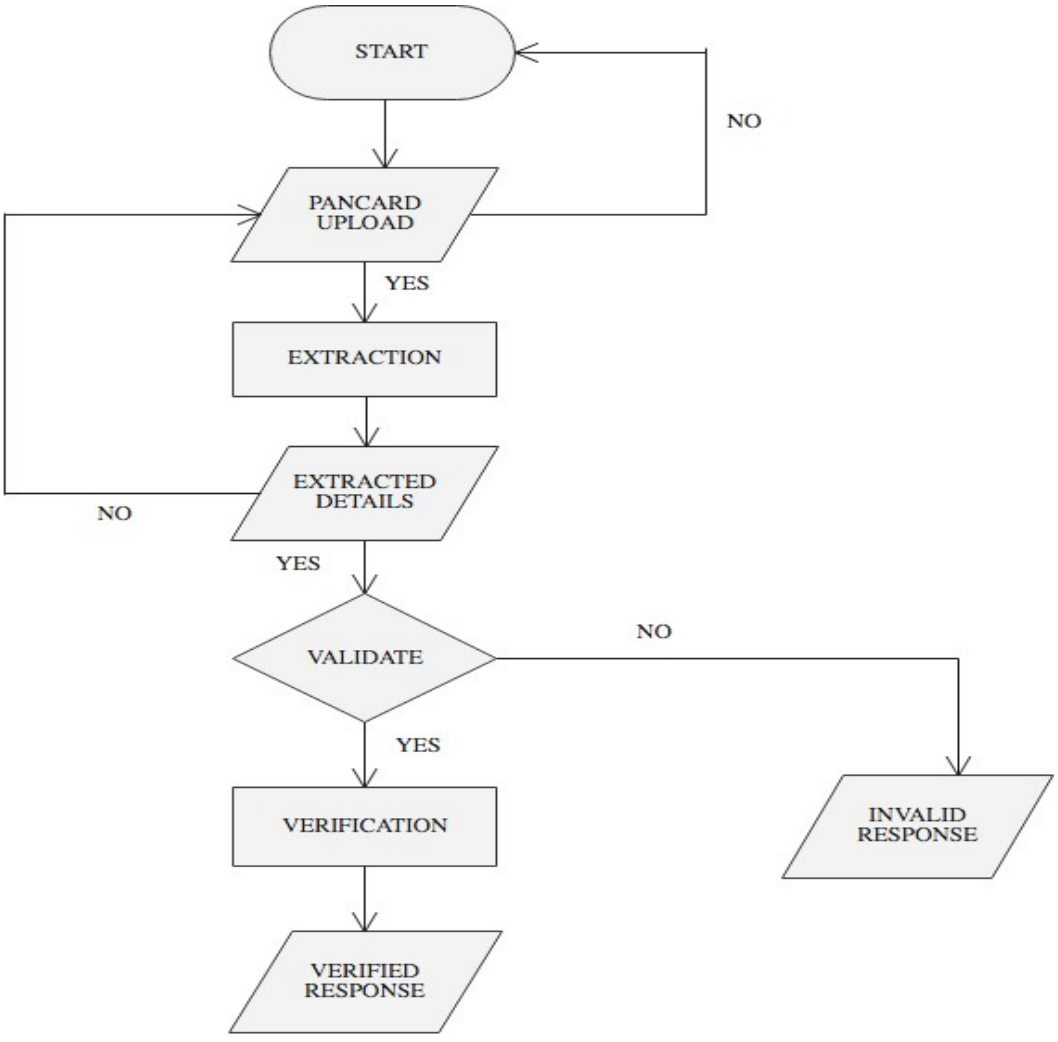


FIG 4.7.1 PANCARD FLOWCHART

4.7.2 AADHAR CARD FLOWCHART

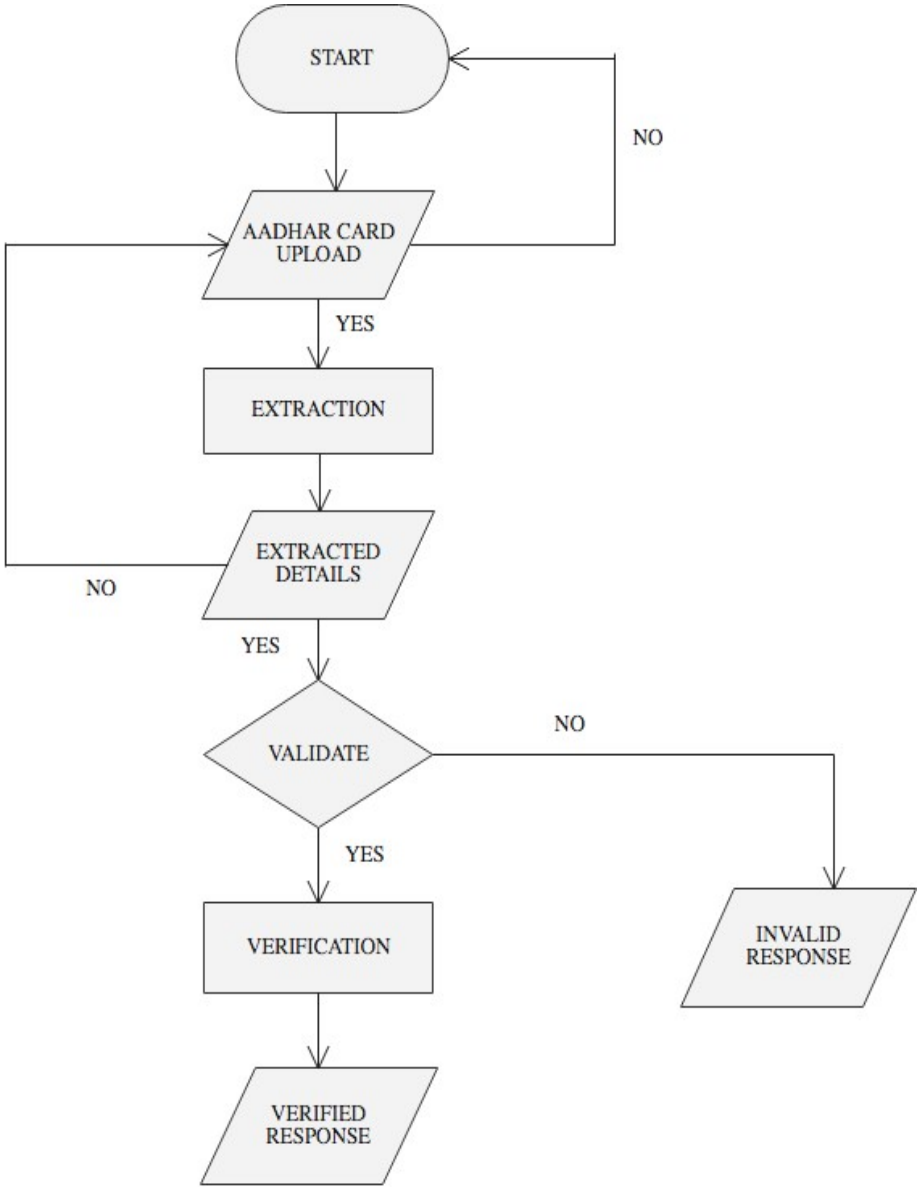


FIG 4.7.3 AADHAR CARD FLOWCHART

4.7.3 DRIVING LICENCE FLOWCHART

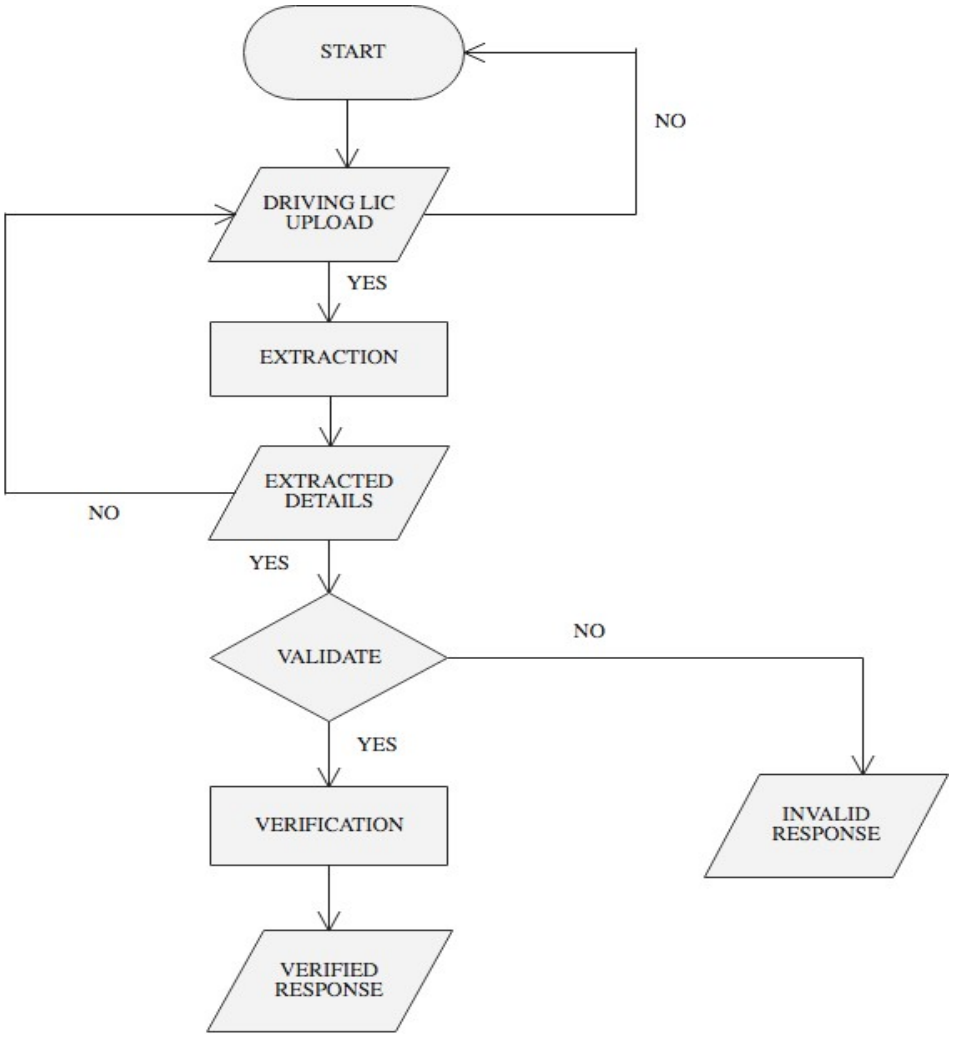


FIG 4.7.3 DRIVING LICENCE FLOW CHART

4.7.4 VOTER ID FLOWCHART

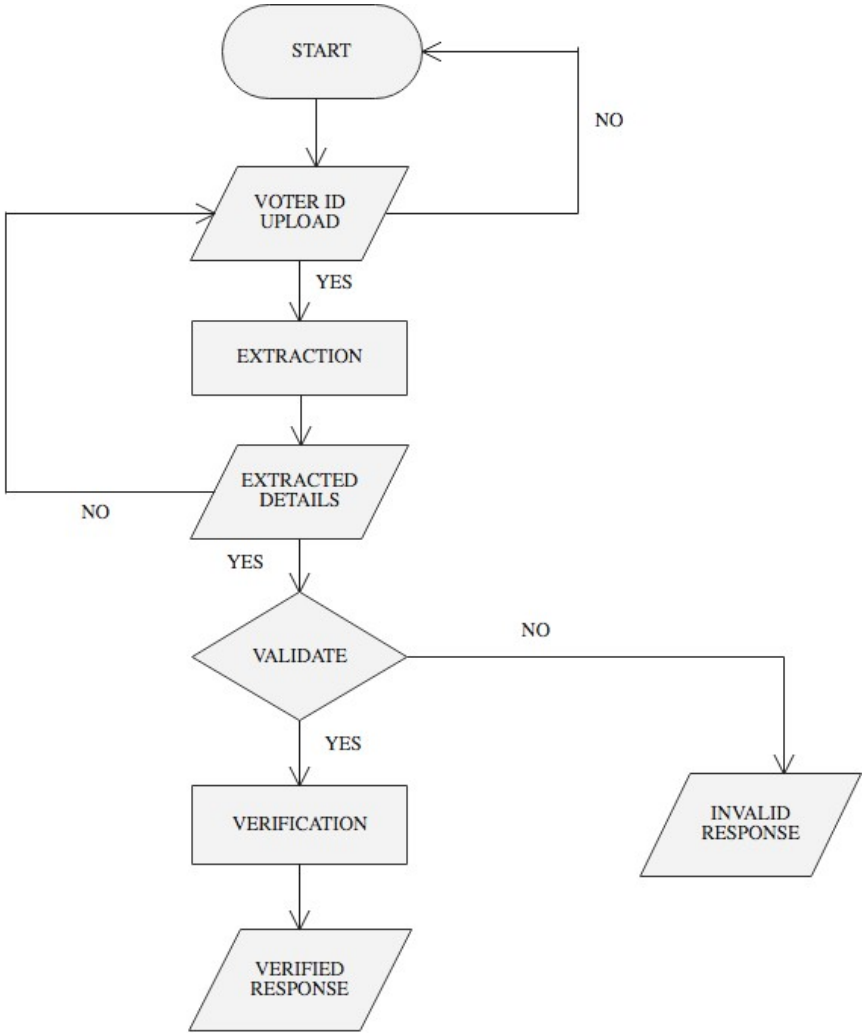


FIG 4.7.4 VOTER ID FLOWCHART

CHAPTER-5
CODING

5.1 FEATURES OF LANGUAGE

Python Overview

Python is a general-purpose, interpreted, high-level programming language which is widely used nowadays .It is an open source language which was developed by Guido Van Rossum in the late 1980s .Python Software Foundation (PSF), a non-profit organization, holds the intellectual property rights of Python. Python was released in 1991 at Centrum Wiskunde & Informatica (CWI) in the Netherlands as a successor to the ABC language. Rossum named this language after a popular comedy show called 'Monty Python's Flying Circus' (and not after Python-the snake).In the last few years, the popularity of python has increased immensely due to its wide applications. According to most of the tech surveys, Python is in the top ten Most Popular Technologies in 2019.

Python Features

Python is an interpreter-based language, which allows execution of one instruction at a time.

a) Extensive basic data types are supported e.g. numbers (floating point, complex, and unlimited-length long integers), strings (both ASCII and Unicode), lists, and dictionaries.
b) Variables can be strongly typed as well as dynamic typed.
c) Supports object-oriented programming concepts such as class, inheritance, objects, module, namespace etc.
d) Cleaner exception handling support.
e) Supports automatic memory management.

Python Advantages

a) Python provides enhanced readability. For that purpose, uniform indents are used to delimit blocks of statements instead of curly brackets, like in many languages such as C, C++ and Java.
b) Python is free and distributed as open-source software. A large programming community is actively involved in the development and support of Python libraries for various applications such as web frameworks, mathematical computing and data science.
c) Python is a cross-platform language. It works equally on different OS platforms like Windows, Linux, Mac OSX etc. Hence Python applications can be easily ported across OS platforms.
d) Python supports multiple programming paradigms including imperative, procedural, object-oriented and functional programming styles.
e) Python is an extensible language. Additional functionality (other than what is provided in the core language) can be made available through modules and packages written in other languages (C, C++, Java etc)
f) A standard DB-API for database connectivity has been defined in Python. It can be enabled using any data source (Oracle, MySQL, SQLite etc.) as a backend to the Python program for storage, retrieval and processing of data.
g) Standard distribution of Python contains the Tkinter GUI toolkit, which is the implementation of popular GUI library called Tcl/Tk. An attractive GUI can be constructed using Tkinter. Many other GUI libraries like Qt, GTK, WxWidgets etc. are also ported to Python.
h) Python can be integrated with other popular programming technologies like C, C++, Java, ActiveX and CORBA.

Python Applications

Even though Python started as a general-purpose programming language with no particular application as its focus, over last few years it has emerged as the language of choice for developers in some application areas. Some important applications of Python are summarized below:

Image Processing

The OpenCV library is commonly used for face detection and gesture recognition. OpenCV is a C++ library, but has been ported to Python. Because of the rapid development of this feature, Python is a very popular choice from image processing.

OpenCV

OpenCV is a cross-platform library using which we can develop real-time computer vision applications. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection. Computer Vision can be defined as a discipline that explains how to reconstruct, interrupt, and understand a 3D scene from its 2D images, in terms of the properties of the structure present in the scene. It deals with modeling and replicating human vision using computer software and hardware.

Django

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source.

Django Features

Ridiculously fast:- Django was designed to help developers take applications from concept to completion as quickly as possible.

Fully loaded:- Django includes dozens of extras you can use to handle common Web development tasks. Django takes care of user authentication, content administration, site maps, RSS feeds, and many more tasks — right out of the box.

Reassuringly secure:- Django takes security seriously and helps developers avoid many common security mistakes, such as SQL injection, cross-site scripting, cross-site request forgery and clickjacking. Its user authentication system provides a secure way to manage user accounts and passwords.

Exceedingly scalable:- Some of the busiest sites on the planet use Django's ability to quickly and flexibly scale to meet the heaviest traffic demands.

Incredibly versatile:- Companies, organizations and governments have used Django to build all sorts of things — from content management systems to social networks to scientific computing platforms.

5.2 FUNCTIONAL DESCRIPTION

```
def ocr(filename):
    """
    This function will handle the core OCR processing of images.
    """
```

```
def clear_text(text):
“””
        it clear up the data extracted .It will make more clarity to letters exrtracted and remove unwanted data

“””
def text_process(text):
“””
        process the extracted text data
“””

def ocr_to_json(text):
“””
        when this function triggers it return data  as dictionary format

“”””””

def  face_detect(filename):
“””
        This function detect the front face in an id card  if there is one.other wise return none

“””

#pancard.py

# from django.conf import settings

"""try:
   from PIL import Image
```

```
except ImportError:
    import Image"""

import pytesseract
import numpy as np
import cv2
import ftfy
import re
import json
import io
# import os
from PIL import Image

# from scipy.ndimage import rotate
face_classifier = cv2.CascadeClassifier("./haarcascade_frontalface_default.xml")

# filename = cv2.imread("/home/arijit/Documents/PAN-Card-OCR-master/media")

def Convert(a):
    it = iter(a)
    res_dct = dict(zip(it, it))
    return res_dct

def ocr(filename):
    """
    This function will handle the core OCR processing of images.
    """
    i = cv2.imread(filename)

    newdata = pytesseract.image_to_osd(i)
    angle = re.search('(?<=Rotate: )\d+', newdata).group(0)
    angle = int(angle)
```

```
    i = Image.open(filename)
    if angle != 0:
        # with Image.open("ro2.jpg") as i:
        rot_angle = 360 - angle
        i = i.rotate(rot_angle, expand="True")
        i.save(filename)

    """rot = False
    if rot:
        i = rotate(i, None)
        i = rotate(i, None)
        i = rotate(i, None)"""
    i = cv2.imread(filename)
    # Convert to gray
    i = cv2.cvtColor(i, cv2.COLOR_BGR2GRAY)

    # Apply dilation and erosion to remove some noise

    kernel = np.ones((1, 1), np.uint8)
    i = cv2.dilate(i, kernel, iterations=1)
    i = cv2.erode(i, kernel, iterations=1)

    text = pytesseract.image_to_string(i)
    # return text



    dict = text.split(' ')
    # print(dict)

    # print('Pan card number is {}.\nDate of Birth is {}.'.format(val[0][0],val[1][0]))
    # print('\n')
    # Cleaning all the gibberish text
```

```
    text = ftfy.fix_text(text)
    text = ftfy.fix_encoding(text)
    new_text = ocr_to_json(text)

     # print(type(new_text))
    # list_text = new_text.split("\n")
    # print(list_text)

    face_detect(filename)
    return new_text

def ocr_to_json(text):
    # Initializing data variable
    name = None
    fname = None
    dob = None
    pan = None

    nameline = []
    dobline = []
    panline = []
    text0 = []
    text1 = []
    text2 = []

    # Searching for PAN
    lines = text.split('\n')
    for lin in lines:
        s = lin.strip()
        s = lin.replace('\n', '')
        s = s.rstrip()
        s = s.lstrip()
```

```
        text1.append(s)

    text1 = list(filter(None, text1))

    # to remove any text read from the image file which lies before the line 'Income Tax
Department'

    lineno = 0  # to start from the first line of the text file.

   for wordline in text1:
        xx = wordline.split('\n')
        if ([w for w in xx if re.search(
     '(INCOMETAXDEPARWENT @|mcommx|INCOME|TAX|GOW|GOVT|
GOVERNMENT|OVERNMENT|VERNMENT|DEPARTMENT|EPARTMENT|
PARTMENT|ARTMENT|INDIA|NDIA)$',
                w)]):
            text1 = list(text1)

            lineno = text1.index(wordline)
            break

    text0 = text1[lineno + 1:]

    # print(text0)  # Contains all the relevant extracted text in form of a list - uncomment
to check

 def findword(textlist, wordstring):
        lineno = -1
        for wordline in textlist:
            xx = wordline.split()
            if ([w for w in xx if re.search(wordstring, w)]):
                lineno = textlist.index(wordline)
```

```
            textlist = textlist[lineno + 1:]
            return textlist
    return textlist


try:
    # Cleaning first names, better accuracy
    name = text0[0]
    name = name.rstrip()
    name = name.lstrip()
    name = name.replace("8", "B")
    name = name.replace("0", "D")
    name = name.replace("6", "G")
    name = name.replace("1", "I")
    name = re.sub('[^a-zA-Z] +', ' ', name)

    # Cleaning Father's name
    fname = text0[1]

    fname = fname.rstrip()
    fname = fname.lstrip()
    fname = fname.replace("8", "S")
    fname = fname.replace("0", "O")
    fname = fname.replace("6", "G")
    fname = fname.replace("1", "I")
    fname = fname.replace("\"", "A")
    fname = re.sub('[^a-zA-Z] +', ' ', fname)

    # Cleaning DOB
    dob = text0[2]
    dob = dob.rstrip()
    dob = dob.lstrip()
    dob = dob.replace('l', '/')
```

```
        dob = dob.replace('L', '/')
        dob = dob.replace('I', '/')
        dob = dob.replace('i', '/')
        dob = dob.replace('|', '/')
        dob = dob.replace('\"', '/1')
        dob = dob.replace(" ", "")

        # Cleaning PAN Card details
        text0 = findword(text1, '(Pormanam|Number|umber|Account|ccount|count|
Permanent|ermanent|manent|wumm)$')
        panline = text0[0]
        pan = panline.rstrip()
        pan = pan.lstrip()
        pan = pan.replace(" ", "")
        pan = pan.replace("\"", "")
        pan = pan.replace(";", "")
        pan = pan.replace("%", "L")

    except:
        pass

        # Making tuples of data
    data = {}
    data['Name'] = name
    data['Father Name'] = fname
    data['Date of Birth'] = dob
    data['PAN'] = pan

    # Writing data into JSON
    try:
        to_unicode = unicode
    except NameError:
```

```
        to_unicode = str

    # Reading data back JSON(give correct path where JSON is stored)
    with open('data.json', 'r', encoding='utf-8') as f:
        ndata = json.load(f)

    t = (f"Name: {data['Name']}\n"
         f"Father's Name: {data['Father Name']}\n"
         f"Date of Birth: {data['Date of Birth']}\n"
         f"PAN number: {data['PAN']}\n")

    list_data = [data['Name'], data['Father Name'], data['Date of Birth'],data['PAN']]

    return list_data
    # return Respone({"message":data})

# print(ocr('images/ocr_example_1.png'))

def  face_detect(filename):
    img = cv2.imread(filename)

    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # cv2.imshow('Original image', img)


 faces = face_classifier.detectMultiScale(gray, 1.3, 5)

    """
    if faces is ():
        print("No faces found")"""
```

```
    # crop_img = 0
    for (x, y, w, h) in faces:
        x = x - 25
        y = y - 40
        cv2.rectangle(img, (x, y), (x + w + 50, y + h + 70), (27, 200, 10), 2)
        # cv2.imshow('Face Detection', img)
        crop_img = img[y: y + h + 70, x: x + w + 50]
        cv2.imwrite('./media/Face1.jpg', crop_img)
        cv2.waitKey(1000)
    cv2.destroyAllWindows()
    return crop_img

#aadhar.py

from django.conf import settings

import pytesseract
import cv2
import re

import cv2
import ftfy
from PIL import Image

# from scipy.ndimage import rotate
import numpy as np

face_classifier = cv2.CascadeClassifier("./haarcascade_frontalface_default.xml")

def adhar(filename):
    """
    This function will handle the core OCR processing of images.
```

```
    i = cv2.imread(filename)
    newdata = pytesseract.image_to_osd(i)
    angle = re.search('(?<=Rotate: )\d+', newdata).group(0)
    angle = int(angle)
    i = Image.open(filename)
    if angle != 0:
        # with Image.open("ro2.jpg") as i:
        rot_angle = 360 – angle

        i = i.rotate(rot_angle, expand="True")
        i.save(filename)

    i = cv2.imread(filename)
    # Convert to gray
    i = cv2.cvtColor(i, cv2.COLOR_BGR2GRAY)

    # Apply dilation and erosion to remove some noise
    kernel = np.ones((1, 1), np.uint8)
    i = cv2.dilate(i, kernel, iterations=1)
    i = cv2.erode(i, kernel, iterations=1)

    text = pytesseract.image_to_string(i)
    # return text
    dict = text.split(' ')
    # print(dict)

    # print('Pan card number is {}.\nDate of Birth is {}.'.format(val[0][0],val[1][0]))
    # print('\n')
    # Cleaning all the gibberish text
    text = ftfy.fix_text(text)
    text = ftfy.fix_encoding(text)
```

```
    new_text = clear_text(text)
    # print(type(new_text))
    face_detect(filename)
    return new_text

def clear_text(text):
    list_text = []
    res = text.split()
    split_ocr = text.split('\n')
    yob_patn = '[0-9]{4}'
    dob_patn = '\d+[-/]\d+[-/]\d+'
    adhar_number_patn = '[0-9]{4}\s[0-9]{4}\s[0-9]{4}'
    adhar_name_patn = r'\b[A-Z][a-z]+\s[A-Z][a-z]+\s[A-Z][a-z]+$'

    # adhar_name_pattrn = r'\b[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}\b'
    name = 'NULL'
    for ele in split_ocr:
        match = re.search(adhar_name_patn, ele)
        if match:
            name = match.group()
            print('name :', name)
            list_text.append(name)
    if name == 'NULL':
        if 'Government' in res:
            index = res.index('India')
            name = res[index + 3] + " " + res[index + 4]
        elif 'GOVERNMENT' in res:
            index = res.index('INDIA')
            name = res[index + 3] + " " + res[index + 4]
        else:
            name = split_ocr[1] + " " + split_ocr[2]
```

```
        print('name :', name)
        list_text.append(name)
    aadhar_number = ''
    for word in res:
        if 'yob' in word.lower():

            yob = re.findall('d+', word)
            if yob:
                print('Year of Birth: ' + yob[0])
                list_text.append(yob[0])
        if len(word) == 4 and word.isdigit():
            aadhar_number = aadhar_number + word + ' '
    if len(aadhar_number) >= 14:
        print("Aadhar number is : " + aadhar_number)
        list_text.append(aadhar_number)
    else:
        aadhar_number = 'NULL'
        print("Aadhar number not read")
    if aadhar_number == 'NULL':
        match = re.search(adhar_number_patn, text)
        print(match, 'match --3')
        if match:
            print(match.group(), 'match.group()')
            aadhar_number = match.group()
            print("Aadhar number is : " + aadhar_number)
            list_text.append(aadhar_number)
    if 'DOB' in text:
        match = re.search(dob_patn, text)
        if match:
            DateOfBirth = match.group()
            print('DateOfBirth :', DateOfBirth)
            list_text.append(DateOfBirth)
```

```
    elif 'DOB:' in res:
        match = re.search(dob_patn, text)
        if match:
            DateOfBirth = match.group()
            print('DateOfBirth :', DateOfBirth)


            list_text.append(DateOfBirth)
    else:
        match = re.search(dob_patn, text)
        if match:
            DateOfBirth = match.group()
            print('DateOfBirth :', DateOfBirth)
            list_text.append(DateOfBirth)
    if 'Year of Birth' in text:
        match = re.search(yob_patn, text)
        if match:
            DateOfBirth = match.group()
            print('DateOfBirth :', DateOfBirth)
            list_text.append(DateOfBirth)
    if 'Male' in text or 'MALE' in text:
        GENDER = 'Male'
    elif 'Female' in text or 'FEMALE' in text:
        GENDER = 'Female'
    else:
        GENDER = 'NAN'
    print('GENDER :', GENDER)
    list_text.append(GENDER)
    return list_text


def face_detect(filename):
    # print(filename, "XXX")
    img = cv2.imread(filename)
```

```
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # cv2.imshow('Original image', img)

    faces = face_classifier.detectMultiScale(gray, 1.3, 5)

    """if faces is ():
        print("No faces found")"""

    for (x, y, w, h) in faces:
        x = x - 25
        y = y - 40
        cv2.rectangle(img, (x, y), (x + w + 50, y + h + 70), (27, 200, 10), 2)
        # cv2.imshow('Face Detection', img)
        crop_img = img[y: y + h + 70, x: x + w + 50]
        cv2.imwrite('./media/Face1.jpg', crop_img)

        cv2.waitKey(1000)
    cv2.destroyAllWindows()
    return crop_img

#dirver.py

import pytesseract
import cv2
import re

import cv2

from PIL import Image
import numpy as np
import regex
```

```
def Convert(a):
    it = iter(a)
    res_dct = dict(zip(it, it))
    return res_dct

def driver_license(filename):
    """
    This function will handle the core OCR processing of images.
    """

    i = cv2.imread(filename)
    newdata = pytesseract.image_to_osd(i)
    angle = re.search('(?<=Rotate: )\d+', newdata).group(0)
    angle = int(angle)
    i = Image.open(filename)
    if angle != 0:
        # with Image.open("ro2.jpg") as i:
        rot_angle = 360 - angle
        i = i.rotate(rot_angle, expand="True")
        i.save(filename)

    i = cv2.imread(filename)
    # Convert to gray
    i = cv2.cvtColor(i, cv2.COLOR_BGR2GRAY)

    # Apply dilation and erosion to remove some noise
    kernel = np.ones((1, 1), np.uint8)
    i = cv2.dilate(i, kernel, iterations=1)
    i = cv2.erode(i, kernel, iterations=1)

    txt = pytesseract.image_to_string(i)
```

```
    print(txt)
    """
    for key in ('Issue Date', 'Licence No\.', 'N', 'Validity\(NT\)'):

        print(regex.findall(fr"(?<={'Licence No'}\s*:\s*)\b[^\n]+", txt,
regex.IGNORECASE))
    """
    text = []
    data = {
        'firstName': None,
        'lastName': None,
        'documentNumber': None,
    }

    c = 0
    print(txt)
    pattern = "(?<=KEY\s*:\s*)\b[^\n]+"
    # Splitting lines
    lines = txt.split('\n')

    for lin in lines:
        c = c + 1
        s = lin.strip()
        s = s.replace('\n', '')
        if s:
            s = s.rstrip()
            s = s.lstrip()
            text.append(s)

            try:
                if re.match(r".*Name|.*name|.*NAME", s):
                    name = re.sub('[^a-zA-Z]+', ' ', s)
```

```
            name = name.replace('Name', '')
            name = name.replace('name', '')
            name = name.replace('NAME', '')

            name = name.replace(':', '')
            name = name.rstrip()
            name = name.lstrip()
            nmlt = name.split(" ")
            data['firstName'] = " ".join(nmlt[:len(nmlt) - 1])
            data['lastName'] = nmlt[-1]
        if re.search(r"[a-zA-Z][a-zA-Z]-\d{13}", s):
            data['documentNumber'] = re.search(r'[a-zA-Z][a-zA-Z]-\d{13}', s)
            data['documentNumber'] = data['documentNumber'].group().replace('-', '')
            if not data['firstName']:
                name = lines[c]
                name = re.sub('[^a-zA-Z]+', ' ', name)
                name = name.rstrip()
                name = name.lstrip()
                nmlt = name.split(" ")
                data['firstName'] = " ".join(nmlt[:len(nmlt) - 1])
                data['lastName'] = nmlt[-1]
        if re.search(r"[a-zA-Z][a-zA-Z]\d{2} \d{11}", s):
            data['documentNumber'] = re.search(r'[a-zA-Z][a-zA-Z]\d{2} \d{11}', s)
            data['documentNumber'] = data['documentNumber'].group().replace(' ', '')
            if not data['firstName']:
                name = lines[c]
                name = re.sub('[^a-zA-Z]+', ' ', name)
                name = name.rstrip()
                name = name.lstrip()
                nmlt = name.split(" ")
                data['firstName'] = " ".join(nmlt[:len(nmlt) - 1])
                data['lastName'] = nmlt[-1]
```

```
            """
            if re.match(r".*DOB|.*dob|.*Dob", s):
                yob = re.sub('[^0-9]+', ' ', s)

                yob = re.search(r'\d\d\d\d', yob)
                data['age'] = datetime.datetime.now().year - int(yob.group())
                """
        except:
            pass
    # new_data = Convert(data)
    # print(new_data)
    list_data = [data['firstName'], data['lastName'], data['documentNumber']]
    return list_data  # data

#voterid.py

from django.conf import settings

"""try:
    from PIL import Image
except ImportError:
    import Image"""
import pytesseract
import numpy as np
import cv2
# import ftfy
import re

# import json
# import io
# import os
```

```
face_classifier = cv2.CascadeClassifier("./haarcascade_frontalface_default.xml")

# filename = cv2.imread("/home/arijitsen/PAN-Card-OCR-master/media")

def Convert(a):
    it = iter(a)
    res_dct = dict(zip(it, it))
    return res_dct

def voterid(filename):
    """
    This function will handle the core OCR processing of images.
    """

    i = cv2.imread(filename)

    # Convert to gray
    i = cv2.cvtColor(i, cv2.COLOR_BGR2GRAY)
    i = cv2.adaptiveThreshold(i, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
                    cv2.THRESH_BINARY, 85, 11)

 # Apply dilation and erosion to remove some noise
    kernel = np.ones((5, 5), np.uint8)
    i = cv2.dilate(i, kernel, iterations=1)
    i = cv2.erode(i, kernel, iterations=1)

    text = pytesseract.image_to_string(i)
    # return text
    dict = text.split(' ')

    # print(dict, "XXX")
    # print('Pan card number is {}.\nDate of Birth is {}.'.format(val[0][0],val[1][0]))
```

```
    # print('\n')
    # Cleaning all the gibberish text
    # text = ftfy.fix_text(text)
    # text = ftfy.fix_encoding(text)
    # new_text = ocr_to_json(text)
    # face_detect(filename)
    # print(type(new_text))
    # list_text = new_text.split("\n")
    # print(list_text)

    new_text = get_name(text)
    face_detect(filename)
    file = open("../TextExtract.txt", "w")
    file.write(text)
    file.close()

    # new_text = text_process()
    return new_text


def get_name(text):
    # Initializing data variable
    name = None
    fname = None
    nameline = []
    text0 = []
    text1 = []
    text2 = []

    # Searching
    lines = text.split('\n')
```

```
    for lin in lines:
        s = lin.strip()
        s = s.rstrip()
        s = s.lstrip()
        text1.append(s)
    # print(text1)
    text1 = list(
        filter(None, text1))  # Attribute has to be converted into a list object before any
additional processing
    # print(text1) #at this operation the new line strings become a list of strings


    lineno = 0  # to start from the first line of the text file.


    for wordline in text1:
        xx = wordline.split('\n')
        if ([w for w in xx if re.search(
                '(ELECTOR|PHOTO|IDENTITY|CARD|ELECTION|COMMISSION|INDIA|
IND|NDIA)$',
                w)]):
            text1 = list(text1)
            lineno = text1.index(wordline)
            break
    # text1 = list(text1)
    text0 = text1[lineno + 1:]
    # print(text0) #Contains all the relevant extracted text in form of a list - uncomment
to check
    try:
        for x in text0:
            for y in x.split():
                # print(x)
                nameline.append(x)
```

```
            break
    except:
        pass
    # print(nameline)
    try:
        name = nameline[2].rsplit(':', 1)[1]
        fname = nameline[4].rsplit(':', 1)[1]

    except:
        pass
    # Making tuples of data
    data = {}
    data['Name'] = name
    data['Father Name'] = fname

    ##
    def findword(textlist, wordstring):
        lineno = -1
        for wordline in textlist:
            xx = wordline.split()
            if ([w for w in xx if re.search(wordstring, w)]):
                lineno = textlist.index(wordline)
                textlist = textlist[lineno + 1:]
                return textlist
        return textlist

    # Finding the electors number
    voter_no = findword(text1, '(ELECTION COMMISSION OF INDIA | ELECTOR
PHOTO IDENTITY CARD|CARD|IDENTITY CARD)$')
    voter_no = voter_no[0]
    epic_no = voter_no.replace(" ", "")
```

```
    # print('\n')
    # print('Epic No:',epic_no)

    ##print(str(d).replace("{","").replace("}", ""))
    print(data['Name'])
    print(data['Father Name'])
    print(voter_no)
    list_data = [data['Name'], data['Father Name']]
    return list_data  # (str(data).replace("{","").replace("}", ""))

def face_detect(filename):
    img = cv2.imread(filename)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # cv2.imshow('Original image', img)

    faces = face_classifier.detectMultiScale(gray, 1.3, 5)

    """if faces is ():
        print("No faces found")"""

    for (x, y, w, h) in faces:
        x = x - 25
        y = y - 40
        cv2.rectangle(img, (x, y), (x + w + 50, y + h + 70), (27, 200, 10), 2)
        # cv2.imshow('Face Detection', img)
        crop_img = img[y: y + h + 70, x: x + w + 50]
        cv2.imwrite('./media/Face2.jpg', crop_img)

    cv2.waitKey(1000)
    cv2.destroyAllWindows()
    return crop_img
```

```
"""
def text_process():

    # Open and reading the textfile containing result
    filename = open('../TextExtract.txt', 'r')
    text = filename.read()

    text1 = []

    # Splitting the lines to sort the text paragraph wise
    lines = text.split('\n')
    for lin in lines:
        s = lin.strip()
        s = s.rstrip()
        s = s.lstrip()
        text1.append(s)

    # Using regex to find the neceesary information
    def findword(textlist, wordstring):
        lineno = -1
        for wordline in textlist:
            xx = wordline.split()
            if ([w for w in xx if re.search(wordstring, w)]):
                lineno = textlist.index(wordline)
                textlist = textlist[lineno+1:]
                return textlist
        return textlist

    # Finding the electors number
    voter_no = findword(text1, '(ELECTION COMMISSION OF INDIA ELECTORAL
PHOTO IDENTITY CARD|CARD|IDENTITY CARD)$')
```

```
    voter_no = voter_no[0]

    epic_no = voter_no.replace(" ", "")
    print('\n')
    print('Epic No:',epic_no)

    # Some voter id's last name is printed on next line hence, it will extract from next line
    find_word = "(Elector's Name|NAME|Name)$"
    name_end = findword(text1, find_word)
    endname = name_end[0]

     lines = text
    for x in lines.split('\n'):
        _ = x.split()
        if ([w for w in _ if re.search("(Elector's Name|ELECTOR'S NAME|NAME|Name|
name)$", w)]):
            person_name = x
            person_name = person_name.split(':')[1].strip()

            # If voter id's endname is on next line it will join it
            if endname:
                print("Elector's Name:",person_name + ' ' + endname)
                full_name = person_name + ' ' + endname
            else:
                print(person_name)
                full_name = person_name

        # Finding the father/husband/mother name
        if ([w for w in _ if re.search("(Father's|Mother's|Husband's)$", w)]):
            elder_name = x
            elder_name = elder_name.split(':')[1].strip()
```

```
            print("Father's Name:",elder_name)

        # Finding the gender of the electoral candidate
        if ([w for w in _ if re.search('(sex|SEX|Sex)$', w)]):
            gender = x
            gender = gender.split('/')
            sex = ''.join(gender[2]).strip()
            print('Sex:',sex)

        # Finding the Date of Birth
        if ([w for w in _ if re.search('(Year|Birth|Date of Birth|DATE OF BIRTH|DOB)$',
w)]):
            year = x
            year = year.split(':')
            dob = ''.join(year[1:]).strip()
            print('Date of Birth:',dob)

    # Converting the extracted informaton into json
    di = {'Epic No':epic_no,
        'Elector Name':full_name,
        'Father Name':elder_name
        #'Sex':sex,
        #'Date of Birth':dob
        }
```

CHAPTER-6
TESTING

Testing is a set of activities that can be planned in advance and conducted systematically. System testing is that stage of implementation that is aimed for ensuring that the system works accurately and efficiently before live operation commences. System testing is the execution of the program to check logical changes and its impact on the output of the system with the intention of finding error. A successful test is the one that uncovers a yet undiscovered error. Testing is vital to the success of the system

A strategy for software testing integrates software test case design methods into well planned series of steps that result in successful completion of the software. A software testing strategy should be flexible enough to promote a customized testing approach. A strategy for testing should accommodate low level test that are necessary to verify implemented as well as high level tests that validate major system functions against customer requirements.

Levels of Testing

Testing is done in the following levels:

a) Unit Testing
b) Integration Testing
c) Validation Testing
d) Output Testing

Unit Testing

In this level of testing each of the modules were tested one by one individually, before they were integrated. The modules were tested as soon as they were developed.

It helped to recognize areas requiring modifications and corrections. In this system, unit testing can be easily accomplished. The project is divided into four modules trainingdata generation, number plate detection, character segmentation and user. The modules includes various activities. In the case of this project, each module was built separately and was tested and verified individually before integrating them to a complete project.

Integration Testing

The modules that are tested individually and confirmed to be working according to specifications are then integrated to form the entire system. The training data generation, number plate detection, character segmentation and user modules are integrated in this testing step and then the entire system is tested. The interface between the modules are thoroughly tested. It is ensured that data and controls are passed properly between modules

Validation Testing

At the end of the integration testing software is completely assembled as a package and interfacing errors have been uncovered and corrected and final series of software validation testing begins

Once the system is integrated and tested to operate properly, it is tested to see if the software developed meets all functional, behavioural and performance requirements. The errors which were uncovered during Integration testing were covered here.

In this system. I have tested the different forms to see whether the inputs given to the forms are stored in the appropriate entries. I also tested the home page with image/video inputs, to see whether the inputs are sent to the system and are validated properly. Hence, I found that the validation testing had carried out successfully.

Output Testing

Here I have tested the system to determine whether it produce the required output or not and to ensure the correctness of the output and its format. In my project testing is implemented in each of the modules. The following test cases are performed to do the testing.

a) Proper opening and closing of window.
b) Appropriate menu bars displayed in appropriate section.
c) A proper working of system is checked for multiple input modes.

Like this all other processes are tested and hence come under the conclusion that there arise no errors. It also found that there arise no modification and corrections in any areas of the project.

CHAPTER -7
IMPLEMENTATION

Implementation is the stage of the project where theoretical design is turned into a working system. If the implementation is not carefully planned and controlled, it can cause chaos and confusion. Proper implementation is essential to provide a reliable system to meet organization requirements. Successful implementation may not guarantee improvement in the organization using the system, but proper installation will prevent it. The process of putting the developed system in actual use is called system implementation. The system can only be implemented after thorough testing is done and if it is found to be working according to the specifications.

The implementation stage involves following tasks:

a) Investigation of system and constraints.

b) Design of methods to achieve the changeover.

c) Training of the staff in the changeover phase.

d) Careful Planning

In order to have the system running of a specific machine, the following components are needed.

a) Python
b) A preferable operating system like windows 8 or windows 10
c) Pycharm
d) Android Phone

Parallel run is done and both the computerized and manual systems are executed in parallel manual result can be compared with the result of computerized system. For the case of demonstration of the success of this system, it was implemented with successfully running; manual systems results are verified.

CHAPTER – 8
CONCLUSION

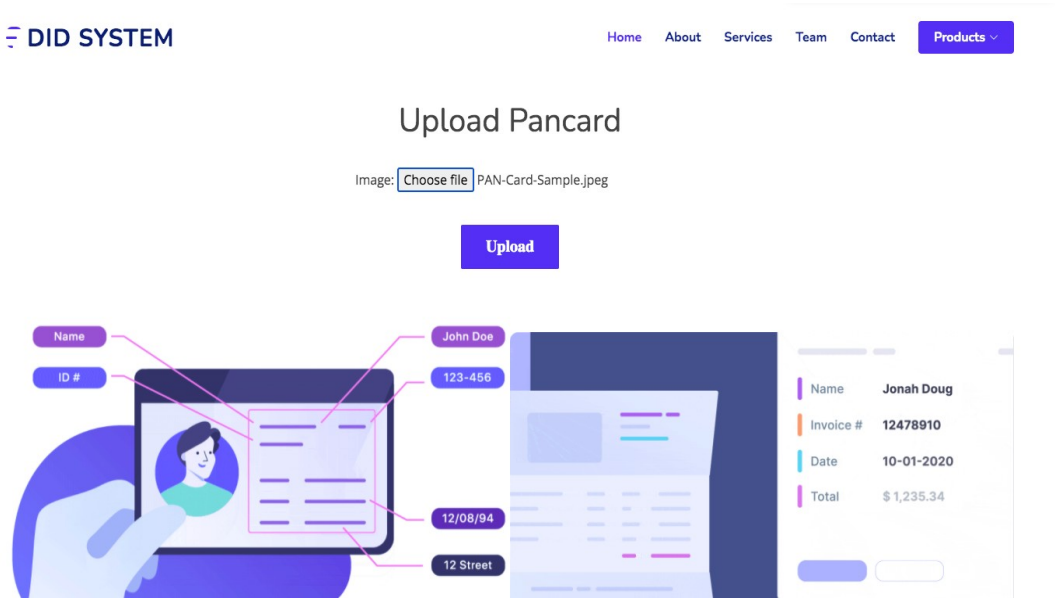
Our DID system uses OCR as one of the first steps in the identity verification process. This step determines whether the identity data is associated with an individual is valid or forged one. DID developed OCR-system that is able to extract characters from identity documents (e.g. aadharcard, pancard, identity cards, and driver's licenses). To do so, DID allows the use of the camera in someone's system, and the use of a scan or photo of an identity document.Both methods face certain industry-wide constraints that must be overcome. Scene complexity may hinder the recognition process, and the challenge is to separate textual parts from non-textual parts of the image. Bad lighting results in dark OCR conditions, while glare results in excessive brightness spots in the image. The challenge is to make sure the camera captures all necessary details, or that the provided scan has all the necessary details. Blurry text may hinder the recognition process, so blur must be prevented as much as possible. Moreover, web applications allow only the use of a single image, so the input image must meet all mandatory requirements for OCR to work correctly. To address these constraints and challenges, in our DID system after each of data extracted it will pass through different functions for cleaning up of data .This technology is able to (1) detect and preprocess the identity document (2) extract the different characters in the preprocessed document by segmentation; and (3) extracted data is passed through function for accurate result .This way, DID system ensures identity verification with higher efficiency and effectiveness than traditional OCR systems.

CHAPTER – 9
FUTURE ENHANCEMENT

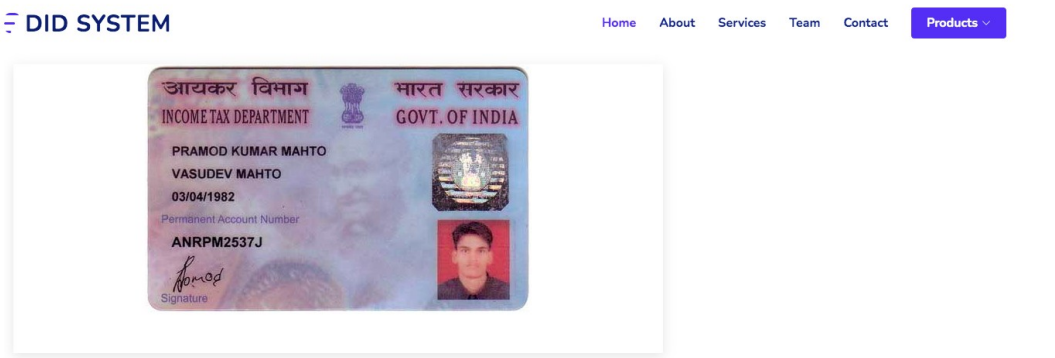
DID system project has a very vast scope in future. The model may be expanded to add AI powered OCR technology assisted and trained by a neural network. In Future it may add biometric verification and real time face recognition so that the verification process could become more effective and accurate

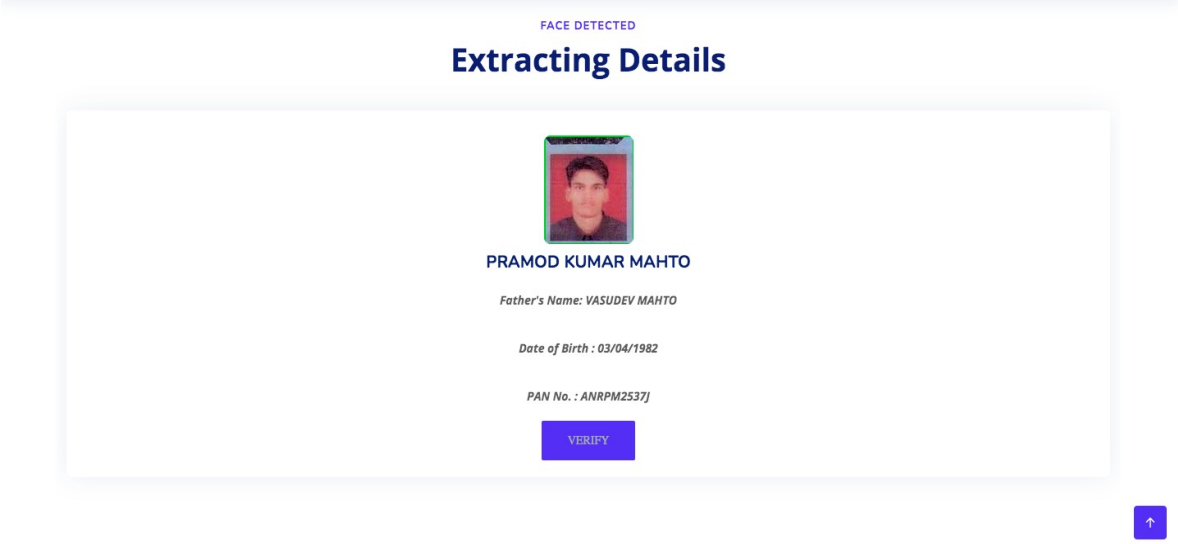
APPENDIX

PAN CARD UPLOAD PAGE

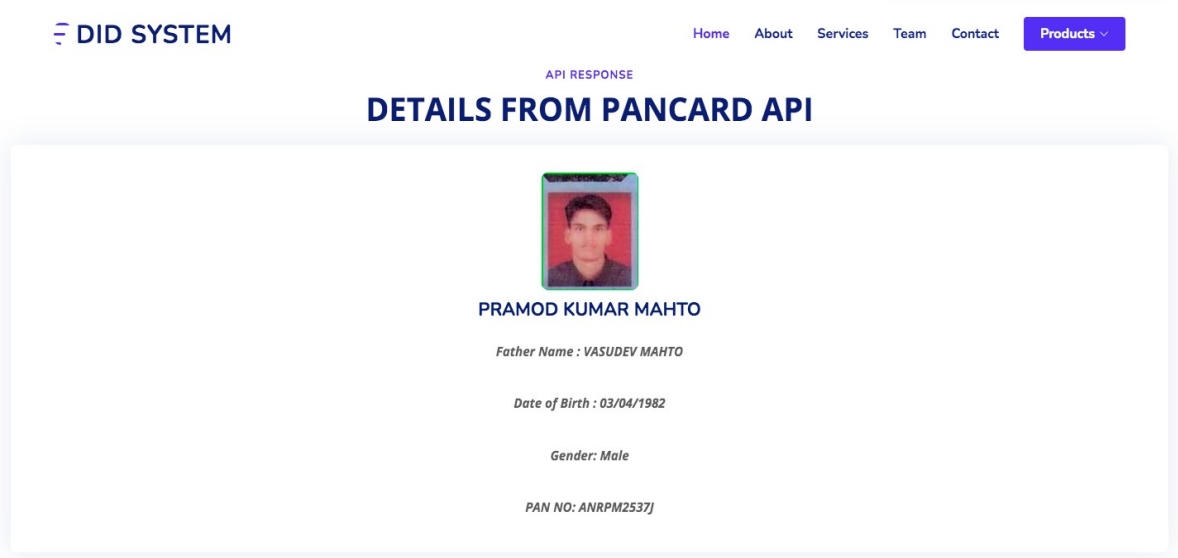


PAN CARD DATA EXTRACT PAGE

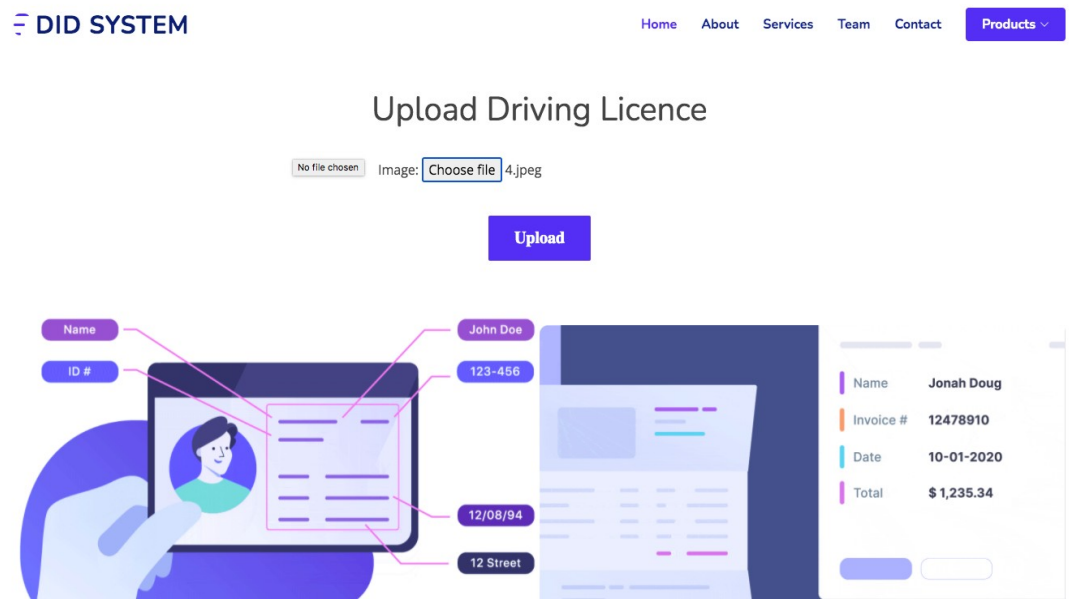




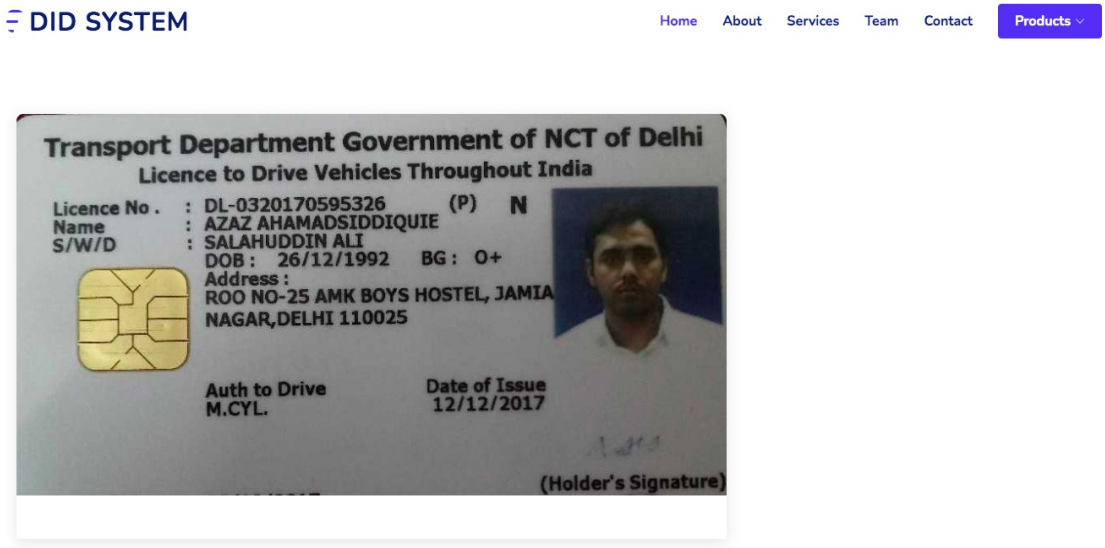
PAN CARD VERIFICATION PAGE

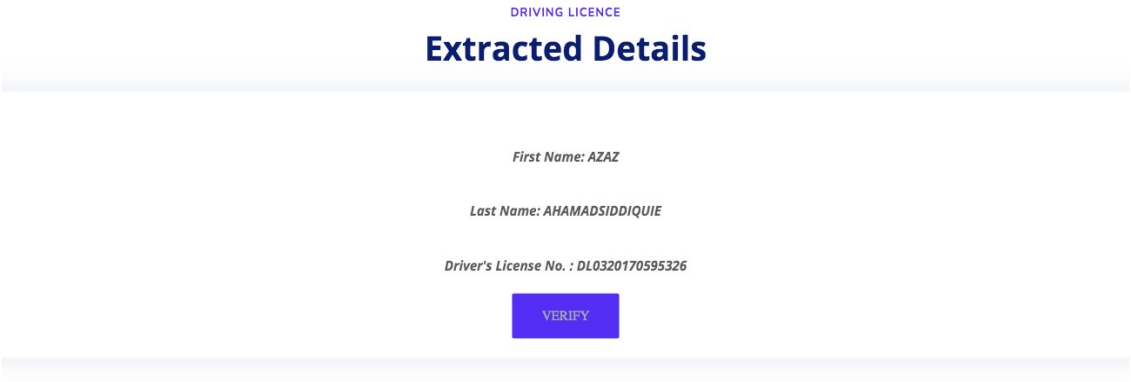


DRIVING LICENCE UPLOAD PAGE

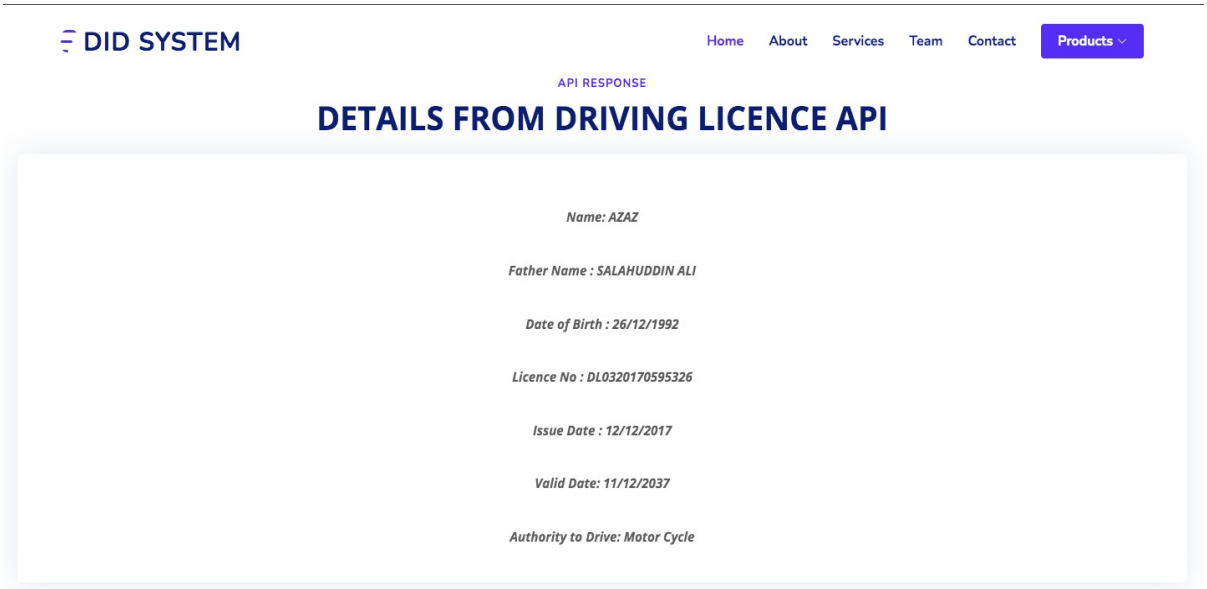


DRIVING LICENCE DATA EXTRACT PAGE





DRIVING LICENCE VERIFICATION PAGE



BIBLIOGRAPHY

WEB URLS

a) https://docs.djangoproject.com/en/3.0/topics/auth/default/#built-in-auth-forms

b) https://www.geeksforgeeks.org/django-url-patterns-python/

c) https://docs.djangoproject.com/en/3.1/topics/templates/

d) https://docs.djangoproject.com/en/3.1/topics/settings/

e) https://docs.djangoproject.com/en/3.1/ref/contrib/staticfiles/

f) https://www.tutorialspoint.com/opencv/index.html

Reference Books

a) OCR with Tesseract, Opencv and python. - Sanjay Gaikwad

b) Two scoops of Django for 1.11 -Daniel Greenfield and Audrey Greenfield

c) High Performance Django -*Peter Baumgartner and Yann Malet*