CHAPTER 1

INTRODUCTION

1.1 Introduction

In the running scene, there is developing interest for the product frameworks to perceive characters in PC framework when data is looked over paper reports as we realize that we have number of papers and books which are in printed configuration identified with various subjects. Nowadays there is an enormous interest in "putting away the data accessible in these paper reports in to a PC stockpiling circle and afterward later reusing this data via looking through procedure". One basic approach to store data in these paper records in to PC framework is to initially examine the archives and afterward store them as IMAGES. Be that as it may, to reuse this data it is extremely hard to peruse the individual substance and looking through the substance structure these records line-by-line and word-by-word. The explanation behind this trouble is the text style qualities of the characters in paper records are distinctive to textual style of the characters in PC framework. Thus, PC can't perceive the characters while understanding them. This idea of putting away the substance of paper reports in PC stockpiling spot and after that perusing and looking through the substance is called DOCUMENT PROCESSING. We in our project will process Aadhar Cards for Automating Student Registration Process.

1.2 Motivation

Automation of fetching and Processing of Data directly from Written or Printed Documents has been getting developed slowly since years. Although not much can be done generally. Hence, we in our project will process Aadhar Cards for Automating Student Registration Process.

1.3 Problem Definition

OCR is a well-known technology which is being used at vast scale at several different scale of economies. OCR is very helpful as a huge amount of data is stored is in image format and it cannot be used directly by the computers and hence OCR allows all those data to be use efficiently by our computers.

In our project, we are leveraging the very advance OCR algorithm to not only extract the data from an image of Aadhar Card but also to classify the data properly . After data extraction and classification, the whole data will be systematically stored in databases and can be reused further for Student Registration purposes.

1.4 Objective of Project

Our Project implement OCR integrated with Document classification in it. Hence the OCR will not only extract the information from the document but understand it accurately. Moreover, all the extracted information will be stored in dedicated databases and all the databases can be searched and scanned for available information. The Extracted Information will be used to Automate Student Registration Process which will make the process Faster.

1.5 Existing System

As mentioned above, OCR technology has been vastly used for extracting information from image documents. OCR is being used for the same operation since a long time. Many OCR algorithms has been used for processing of Bank Cheques which strictly follows a particular format, which is not the case in Aadhar Card.

1.6 Disadvantages of Existing System

- Older Systems are less accurate.
- They cannot understand type of data.
- They cannot classify the data.

1.7 Proposed System

Our Project implement OCR integrated with Document classification in it. Hence the OCR will not only extract the information from the document but understand it accurately. Moreover, all the extracted information will be stored in dedicated databases and all the databases can be searched and scanned for available information. The Extracted Information will be used to Automate Student Registration Process which will make the process Faster.

1.8 Advantages of Proposed System

- Achieving more accuracy and capability to process High Definition images.
- Able to classify the type of document given to extract information automatically.
- Able to categorize the type of information.

CHAPTER 2

REQUIREMENT ANALYSIS

2.1 Requirement Analysis

The project involved analyzing the design of few applications so as to make the application more users friendly. To do so, it was really important to keep the navigations from one screen to the other well ordered and at the same time reducing the amount of typing the user needs to do. In order to make the application more accessible, the browser version had to be chosen so that it is compatible with most of the Browsers.

2.2 Requirement Specification

2.2.1 Functional requirement

In software engineering, a functional requirement defines a system or its component. It describes the functions a software must perform. A function is nothing but inputs, its behavior, and outputs. It can be a calculation, data manipulation, business process, user interaction, or any other specific functionality which defines what function a system is likely to perform.

Functional software requirements help you to capture the intended behavior of the system. This behavior may be expressed as functions, services or tasks or which system is required to perform.

2.2.2 Non-Functional requirement

A non-functional requirement defines the quality attribute of a software system. They represent a set of standards used to judge the specific operation of a system. Example, how fast does the website load?

A non-functional requirement is essential to ensure the usability and effectiveness of the entire software system. Failing to meet non-functional requirements can result in systems that fail to satisfy user needs.

2.3 Computational resource requirements

2.3.1 Hardware requirements:-

- ✓ Processor : i3 or higher
- ✓ RAM : 4 GB
- ✓ Hard Disk : 40 GB
- ✓ Mouse : Optical Mouse.
- ✓ Monitor : 15’ Colour Monitor.

2.3.2 Software requirements:-

- ✓ Operating System : Windows 10 or Above
- ✓ Compiler / Interpreter : Python 3.8
- ✓ Modules : numpy, pandas, tensorflow, cv2, pytesseract, nltk,

CHAPTER 3

DESIGN

3.1 Introduction

This chapter provides the design phase of the Application. System Architecture and To design the project, we use the UML diagrams. The Unified Modelling Language (UML) is a general- purpose, developmental, modelling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system.

3.2 System Architecture

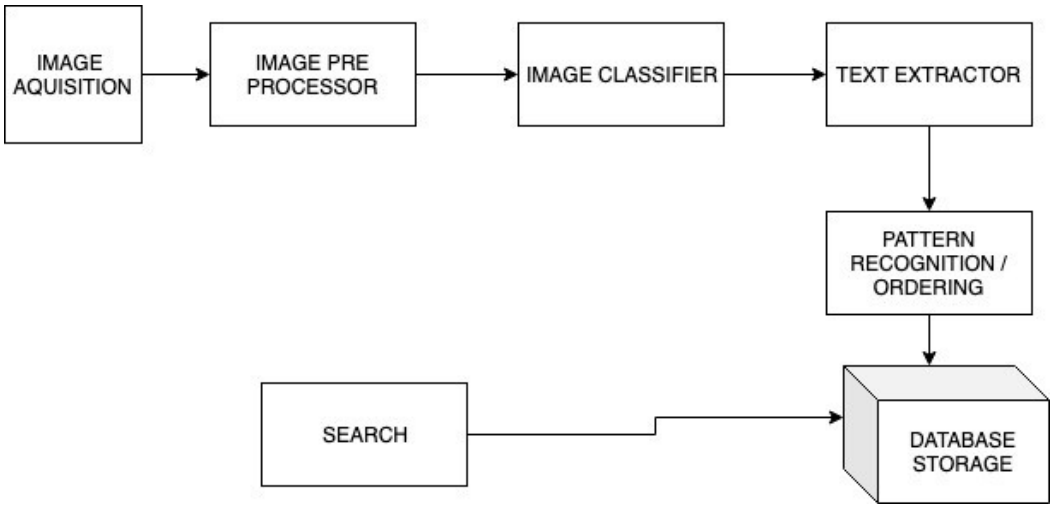

Figure 3.1 System Architecture

The above System Architecture diagram describe about Architecture of project

3.3 UML Diagrams

A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

3.3.1 Class Diagram

The class diagrams are widely used in the modeling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages. Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints.

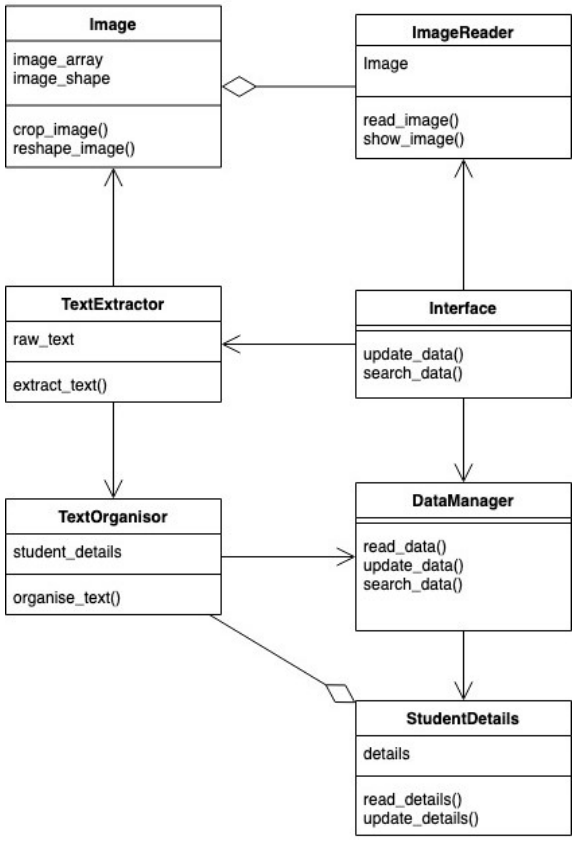

Figure 3.2 Class Diagram

The above Class Diagram represent about class diagram of project

3.3.2 Sequence Diagram

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

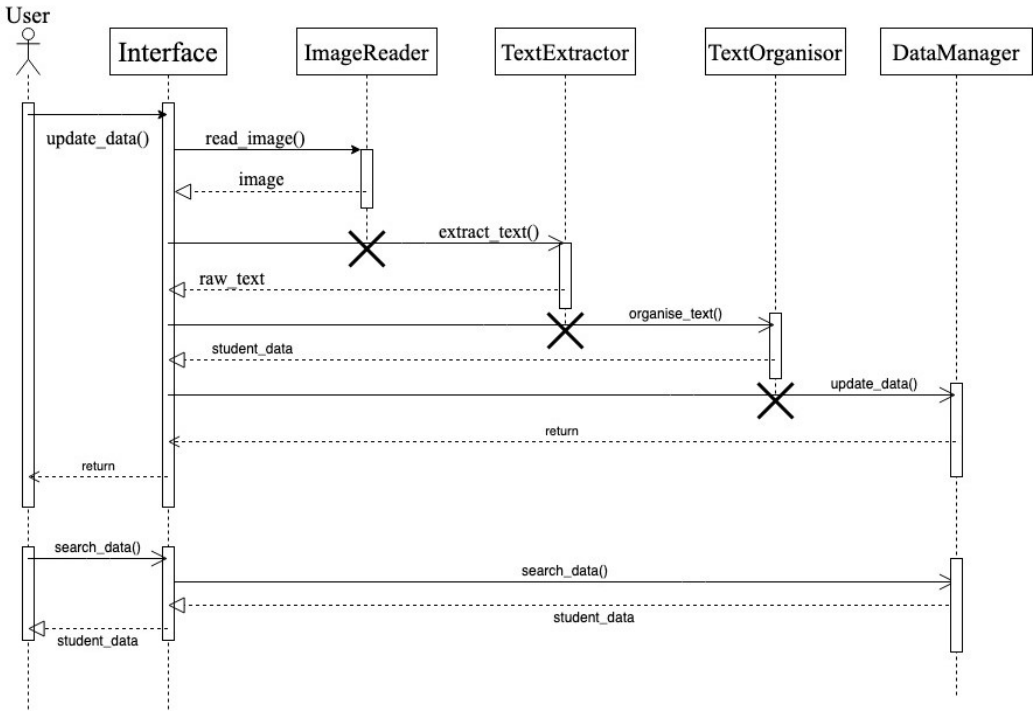

Figure 3.3 Sequence Diagram

The above Sequence Diagram represent about class diagram of project

CHAPTER 4

ALGORITHM & MODULE

4.1 Algorithm

An algorithm is a finite sequence of well-defined, computer-implementable instructions, typically to solve a class of specific problems or to perform a computation. Algorithms are always unambiguous and are used as specifications for performing calculations, data processing, automated reasoning, and other tasks. In contrast, a heuristic is a technique used in problem solving that uses practical methods and/or various estimates in order to produce solutions that may not be optimal but are sufficient given the circumstances.

4.1.1 Update Data Algorithm

Step 1:- Take a snap of Aadhar Card / Driving License.

Step 2:- Classify image with one of the ID's

Step 3:- Extract Text from Image.

Step 4:- Arrange the Text in respective fields.

Step 5:- Update the details in Data base.

4.1.2 Search Data Algorithm

Step 1:- Read the given input

Step 2:- Search Database for Details

Step 3:- Display Details

4.2. Flowchart

Flowchart is a diagram that depicts a process, system or computer algorithm. They are widely used in multiple fields to document, study, plan, improve and communicate often complex processes in clear, easy-to-understand diagrams.

4.2.1 Update Data Flowchart

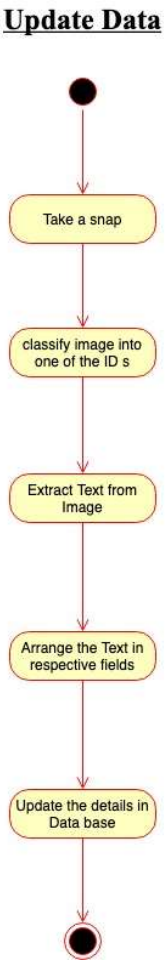

Figure No 2.1 Update Data Flowchart Diagram

The above Figure is represtation of Update Data Flowchart Diagream

4.2.2 Search Data Flowchart

Figure 2.2 Search Data Flowchart

The above Figure is represtation of Search Data Flowchart Diagream

4.3 Modules

Modules are files that contain Python definitions and declarations. Modules can define functions, classes, and variables. Modules can also include executable code. Grouping related code in the module can make it easier to understand and use the code. It also makes the code logical. In Python programming, treat modules as the same as code libraries.

4.3.1 Numpy

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

At the core of the NumPy package, is the *ndarray* object. This encapsulates *n*-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance. There are several important differences between NumPy arrays and the standard Python sequences:

- NumPy arrays have a fixed size at creation, unlike Python lists (which can grow dynamically). Changing the size of an *ndarray* will create a new array and delete the original.

- The elements in a NumPy array are all required to be of the same data type, and thus will be the same size in memory. The exception: one can have arrays of (Python, including NumPy) objects, thereby allowing for arrays of different sized elements.

- NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python's built-in sequences.

- A growing plethora of scientific and mathematical Python-based packages are using NumPy arrays; though these typically support Python-sequence input, they convert such input to NumPy arrays prior to processing, and they often output NumPy arrays. In other words, in order to efficiently use much (perhaps even most) of today's scientific/mathematical Python-based software, just knowing how to use Python's built-in sequence types is insufficient - one also needs to know how to use NumPy arrays.

Vectorization describes the absence of any explicit looping, indexing, etc., in the code - these things are taking place, of course, just "behind the scenes" in optimized, pre-compiled C code. Vectored code has many advantages, among which are:

- vectorized code is more concise and easier to read
- fewer lines of code generally means fewer bugs
- the code more closely resembles standard mathematical notation (making it easier, typically, to correctly code mathematical constructs)
- Vectorization results in more "Pythonic" code. Without vectorization, our code would be littered with inefficient and difficult to read for loops.

4.3.2 Pandas

Pandas is an open source Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named Numpy, which provides support for multi-dimensional arrays. As one of the most popular data wrangling packages, Pandas works well with many other data science modules inside the Python ecosystem, and is typically included in every Python distribution, from those that come with your operating system to commercial vendor distributions like ActiveState's ActivePython.

Pandas makes it simple to do many of the time consuming, repetitive tasks associated with working with data, including:

- Data cleansing
- Data fill
- Data normalization
- Merges and joins
- Data visualization
- Statistical analysis
- Data inspection
- Loading and saving data

- And much more

In fact, with Pandas, you can do everything that makes world-leading data scientists vote Pandas as the best data analysis and manipulation tool available.

4.3.3 TensorFlow

TensorFlow is an open source library for fast numerical computing.

It was created and is maintained by Google and released under the Apache 2.0 open source license. The API is nominally for the Python programming language, although there is access to the underlying C++ API.

Unlike other numerical libraries intended for use in Deep Learning like Theano, TensorFlow was designed for use both in research and development and in production systems, not least RankBrain in Google search and the fun DeepDream project. It can run on single CPU systems, GPUs as well as mobile devices and large scale distributed systems of hundreds of machines.

TensorFlow is a Python library for fast numerical computing created and released by Google.

It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensorFlow.
In this post you will discover the TensorFlow library for Deep Learning.

4.3.4 Opencv-Python

OpenCV-Python is a library of Python bindings designed to solve computer vision problems. Python is a general purpose programming language started by Guido van Rossum that became very popular very quickly, mainly because of its simplicity and code readability. It enables the programmer to express ideas in fewer lines of code without reducing readability.

Compared to languages like C/C++, Python is slower. That said, Python can be easily extended with C/C++, which allows us to write computationally intensive code in C/C++ and create Python wrappers that can be used as Python modules. This gives us two advantages: first, the code is as fast as the original C/C++ code (since it is the actual C++ code working in background) and second, it easier to code in Python than C/C++. OpenCV-Python is a Python wrapper for the original OpenCV C++ implementation.

OpenCV-Python makes use of Numpy, which is a highly optimized library for numerical operations with a MATLAB-style syntax. All the OpenCV array structures are converted to and from Numpy arrays. This also makes it easier to integrate with other libraries that use Numpy such as SciPy and Matplotlib.

Since OpenCV is an open source initiative, all are welcome to make contributions to the library, documentation, and tutorials. If you find any mistake in this tutorial (from a small spelling mistake to an egregious error in code or concept), feel free to correct it by cloning OpenCV in GitHub and submitting a pull request. OpenCV developers will check your pull request, give you important feedback and (once it passes the approval of the reviewer) it will be merged into OpenCV.

4.3.5 Pytesseract

Python-tesseract is an optical character recognition (OCR) tool for python. That is, it will recognize and "read" the text embedded in images.

Python-tesseract is a wrapper for Google's Tesseract-OCR Engine. It is also useful as a stand-alone invocation script to tesseract, as it can read all image types supported by the Pillow and Leptonica imaging libraries, including jpeg, png, gif, bmp, tiff, and others. Additionally, if used as a script, Python-tesseract will print the recognized text instead of writing it to a file.

Functions

- get_languages Returns all currently supported languages by Tesseract OCR..
- get_tesseract_version Returns the Tesseract version installed in the system.
- image_to_string Returns unmodified output as string from Tesseract OCR processing
- image_to_boxes Returns result containing recognized characters and their box boundaries
- image_to_data Returns result containing box boundaries, confidences, and other information. Requires Tesseract 3.05+. For more information, please check the Tesseract TSV documentation
- image_to_osd Returns result containing information about orientation and script detection.
- image_to_alto_xml Returns result in the form of Tesseract's ALTO XML format.
- run_and_get_output Returns the raw output from Tesseract OCR. Gives a bit more control over the parameters that are sent to tesseract.

4.3.6 Nltk

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

Thanks to a hands-on guide introducing programming fundamentals alongside topics in computational linguistics, plus comprehensive API documentation, NLTK is suitable for linguists, engineers, students, educators, researchers, and industry users alike. NLTK is available for Windows, Mac OS X, and Linux. Best of all, NLTK is a free, open source, community-driven project.

NLTK has been called “a wonderful tool for teaching, and working in, computational linguistics using Python,” and “an amazing library to play with natural language.”

Natural Language Processing with Python provides a practical introduction to programming for language processing. Written by the creators of NLTK, it guides the reader through the fundamentals of writing Python programs, working with corpora, categorizing text, analyzing linguistic structure, and more. The online version of the book has been been updated for Python 3 and NLTK 3.

4.3.7 SQLite3

SQLite3 is a very easy to use database engine. It is self-contained, serverless, zero-configuration and transactional. It is very fast and lightweight, and the entire database is stored in a single disk file. It is used in a lot of applications as internal data storage. The Python Standard Library includes a module called "sqlite3" intended for working with this database. This module is a SQL interface compliant with the DB-API 2.0 specification.

Following are important sqlite3 module routines, which can suffice your requirement to work with SQLite database from your Python program. If you are looking for a more sophisticated application, then you can look into Python sqlite3 module's official documentation.

sqlite3.connect(database [,timeout ,other optional arguments])

This API opens a connection to the SQLite database file. You can use ":memory:" to open a database connection to a database that resides in RAM instead of on disk. If database is opened successfully, it returns a connection object.

When a database is accessed by multiple connections, and one of the processes modifies the database, the SQLite database is locked until that transaction is committed. The timeout parameter specifies how long the connection should wait for the lock to go away until raising an exception. The default for the timeout parameter is 5.0 (five seconds).

If the given database name does not exist then this call will create the database. You can specify filename with the required path as well if you want to create a database anywhere else except in the current directory.

connection.cursor([cursorClass])

This routine creates a cursor which will be used throughout of your database programming with Python. This method accepts a single optional parameter cursorClass. If supplied, this must be a custom cursor class that extends sqlite3.Cursor.

cursor.execute(sql [, optional parameters])

This routine executes an SQL statement. The SQL statement may be parameterized (i. e. placeholders instead of SQL literals). The sqlite3 module supports two kinds of placeholders: question marks and named placeholders (named style).

For example − cursor.execute("insert into people values (?, ?)", (who, age))

connection.execute(sql [, optional parameters])

This routine is a shortcut of the above execute method provided by the cursor object and it creates an intermediate cursor object by calling the cursor method, then calls the cursor's execute method with the parameters given.

cursor.executemany(sql, seq_of_parameters)

This routine executes an SQL command against all parameter sequences or mappings found in the sequence sql.

connection.executemany(sql[, parameters])

This routine is a shortcut that creates an intermediate cursor object by calling the cursor method, then calls the cursor.s executemany method with the parameters given.

cursor.executescript(sql_script)

This routine executes multiple SQL statements at once provided in the form of script. It issues a COMMIT statement first, then executes the SQL script it gets as a parameter. All the SQL statements should be separated by a semi colon (;).

connection.executescript(sql_script)

This routine is a shortcut that creates an intermediate cursor object by calling the cursor method, then calls the cursor's executescript method with the parameters given.

connection.total_changes()

This routine returns the total number of database rows that have been modified, inserted, or deleted since the database connection was opened.

connection.commit()

This method commits the current transaction. If you don't call this method, anything you did since the last call to commit() is not visible from other database connections.

connection.rollback()

This method rolls back any changes to the database since the last call to commit().

connection.close()

This method closes the database connection. Note that this does not automatically call commit(). If you just close your database connection without calling commit() first, your changes will be lost!

CHAPTER 5

IMPLEMENTATION

5.1 Introduction

The implementation phase involves python and it's modules to perform the above source code. The Source code describes the program for execution for achieving the aadhar card and driving licensee's data extraction in database by using various features of python and OCR supporting modules.

5.2 Explanation of Key Functions

Extract data:

This Function is responsible for extracting the Data from the Aadhar Image

5.3 Source Code

Main.py

```
import tkinter as tk
from tkinter.filedialog import askopenfilenames
import re
import backend
main_frame = tk.Tk()
main_frame.title("Template recognition")
main_frame.geometry("550x220+30+30")
class GUI:
        def upload(self, main_frame):
```

```
        root = tk.Frame(main_frame,bg = "cyan")

        root.place(x=0,y=0,width = 550, height = 220)

        tk.Label(root,text = "Upload Document To DataBase",fg= "blue",bg =
            "pink",font=("monaco",22,"bold")).place(x = 25,y = 20)

        tk.Label(root,text = "Select the Document",bg =
            "cyan",font=("monaco",10,"bold")).place(x = 10,y= 120)

        self._file = tk.Entry(root, font=("monaco",10,"bold"))

        self._file.place(x = 190, y=120,width=150)

        tk.Button(root, text = "select file", bg = "white",fg =
            "green",font=("monaco",10,"bold"), command = lambda
                :self.fileSelect(self._file)).place(x = 340,y= 119,width =
                110,height=23)

        upload_button = tk.Button(root,text
            ="Upload",font=("monaco",20,"bold"),bg = "white",fg =
                "green",command = lambda


        :backend.data_extract(self.file_address,self._file))

        upload_button.place(x = 390,y = 160, height = 50, width = 120)

        retrieve_button = tk.Button(root,text
            ="Retrieve",font=("monaco",10,"bold"),bg = "white",fg =
                "green",command = lambda :self.retrieve(root))

        retrieve_button.place(x = 10,y = 180, height = 30, width = 100)


def retrieve(self,frame):

        frame.destroy()

        frame = tk.Frame(main_frame,bg = "cyan")

        frame.place(x=0,y=0,width = 550, height = 220)
```

```
        tk.Label(frame,text = "Retrieve Information from Database",fg=
        "blue",bg= "pink",font=("monaco",18,"bold")).place(x = 20,y = 10)

        tk.Label(frame,text = "Enter Your Name",bg =
                "cyan",font=("monaco",10,"bold")).place(x = 10,y= 80)

        name_entry = tk.Entry(frame,font=("monaco",10,"bold"))

        name_entry.place(x = 160,y=80,width=300)

        retrieve_button = tk.Button(frame,text
                ="Retrieve",font=("monaco",20,"bold"),bg = "white",fg =
                "green",                                command = lambda
                :backend.data_retrieve(name_entry))#,command =
                lambda
                :signup(userid,password,name,frame,main_frame,login_gui))

        retrieve_button.place(x = 390,y = 120, height = 70, width = 150)

        Upload_button = tk.Button(frame,text
                ="Upload",font=("monaco",10,"bold"),bg = "white",fg =
                        "green",command = lambda :self.upload(main_frame))

        Upload_button.place(x = 10,y = 180, height = 30, width = 100)


def fileSelect(self,field):

        data = askopenfilenames()

        self.file_address = data

        list_of_files = []

        for i in data:

                i = i.split("/")

                list_of_files.append(i[-1])

        data = ",".join(list_of_files)

        print(data)
```

```
                field.delete(0,"end")

                field.insert(0,data)

gui = GUI()

gui.upload(main_frame)

main_frame.mainloop()
```

Backend.py

```
import tkinter as tk

import pytesseract

import sqlite3

import cv2

import re

import numpy as np


db = sqlite3.connect("userdb")

db.execute('create table if not exists aadhaar(name text primary key not null, address
text, dob text, aadhaarnum text)')

db.execute('create table if not exists licence(name text primary key not null, father
text, address text, licencenum text)')

# data = db.execute("select * from sqlite_master")

def data_extract(file_name_list,field_text):

    field_text.delete(0,"end")

    for file_name in file_name_list:

        try:

            img = cv2.imread(file_name,0)
```

```
data = pytesseract.image_to_string(img)
if "aadhaar" in data.lower() or "government" in data.lower():
    data = pytesseract.image_to_string(img)
    # print(data)
    try:
        # Aadhaar Extraction
        # ----------------------------------------------------------------------------------
        # name extraction
        count = 0
        for i in data.split("\n\n"):
            name = 0
            if "dob" in i.lower() and "dob" not in i.split("\n")[0].lower():
                name = "".join(i.split("\n")[0])
                dob_data = i
                break
            elif "dob" in i.lower() and "dob" in i.split("\n")[0].lower():
                # print("---------------------------")
                dob_data = i
                break
            count+=1
        if name == 0:
            name = data.split("\n\n")[count-1]
        #----------------------------------------------------------------------------------
        # dob extraction
```

```
        dob = re.search("dob: ",dob_data.lower())
        dob = dob_data[dob.end():dob.end()+10]
        # ---------------------------------------------------------------------------------
        # address extraction
        # print(data)
        loc_start = re.search(r"[a-zA-Z]/[oO]",data)
        if loc_start:
            loc_start = loc_start.start()
            print("hello")
            loc_end = re.search(r"\d{6}",data).end()
            address = data[loc_start:loc_end]
        else:
            loc_start = re.search(r"address", data.lower()).end()
            ad_data = "\n".join(data[loc_start+2:].split("\n")[:3])
            address = ad_data.strip()
            # print(ad_data)
        # ---------------------------------------------------------------------------------
        # aadhaar number extraction
        adhar_num = re.search(r"\d{4}\s\d{4}\s\d{4}",data).group()
        # inserting
        db.execute('insert into aadhaar(name,address,dob,aadhaarnum)
    values("{}","{}","{}","{}")'.format(name,address,dob,adhar_num))
        db.commit()
        field_text.insert(0,"Aadhaar Upload Successful")
```

```
        except Exception as e:
            print(e)
            field_text.insert(0,"Aadhaar already exist")
    elif "driving" in data.lower() or "license" in data.lower():
        data = pytesseract.image_to_string(img)
        # driving licence data extraction
        # # ----------------------------------------------------------------------------------
        # # license number
        loc_data = re.search(r"TS\w+\n",data)
        license_number = loc_data.group().strip()


        # # ----------------------------------------------------------------------------------
        # name
        name = []
        for i in data[loc_data.start():].split("\n")[1:]:
            if len(str(i)) != 0:
                name.append(i)
            if len(name) == 2:
                break
        # # ----------------------------------------------------------------------------------
        # address
        if name[0] == name[1]:
            data_s = data[re.search(name[1],data).end():]
            # print(data_s)
```

```
else:
    data_s = data
add_data = re.search(name[1],data_s)
add_data = data_s[add_data.end():].split("\n")[:8]
address = (" ".join(add_data)).strip()
# # ----------------------------------------------------------------------------------
# # issued date
issued_data = ""
for i in range(1,6):
    if len(data.split("\n")[-i]) == 0:
        continue
    issued_data = data.split("\n")[-i]
    break
# issued_date = re.search(r"\d+/\d+/\d+",issued_data).group()
# # # ----------------------------------------------------------------------------------
# Rto = issued_data.split(issued_date)[-1].strip()
# db.execute('create table licence(name text primary key not null, address
      text, licencenum text, issued_date text, rto text)')
# inserting
try:
    db.execute('insert into licence(name, father, address, licencenum)
              values("{}","{}","{}","{}")'.format(name[0], name[1], address,
                    license_number))
    db.commit()
    field_text.insert(0,"DL Upload Successful")
```

```
            except Exception as e:
                print(e)
                field_text.insert(0,"DL data already exist")
        except Exception as e:
            print(e)
            field_text.delete(0,"end")
            field_text.insert(0,"Unknown File Format")
def data_retrieve(widget):
    text = widget.get() #to get the user query
    relevant_aadhaar_data = []
    relevant_licence_data = []
    aadhaar_data = db.execute("select name,address,dob,aadhaarnum from
aadhaar").fetchall()
    licence_data = db.execute("select name, father, address, licencenum from
licence").fetchall()

    for i in aadhaar_data:
        if text.lower() in i[0].lower() or text.lower() in i[2].lower():
            relevant_aadhaar_data.append(i)
    for i in licence_data:
        if text.lower() in i[0].lower():
            relevant_licence_data.append(i)
    print_adhaar_data = '''----------Aadhaar Data----------
name: {}
address: {}
```

```
dob: {}

aadhaar number: {}

          ------------------------------------------------------'''

   print_licence_data = '''----------------------Licence Data----------------------

name: {}

father name: {}

address: {}

licence number: {}

   '''

   if len(relevant_aadhaar_data) == 0:

      print_adhaar_data = "No Aaddhaar data Found\n----------------------------------------
----      ----------------------\n"

   else:

      print(relevant_aadhaar_data)

      print_adhaar_data = print_adhaar_data.format(relevant_aadhaar_data[0][0],
         relevant_aadhaar_data[0][1], relevant_aadhaar_data[0][2],
         relevant_aadhaar_data[0][3])

   if len(relevant_licence_data) == 0:

      print_licence_data = "No licence data Found\n----------------------------------------
-----      --------------------\n"

   else:

      print(relevant_licence_data)

      print_licence_data = print_licence_data.format(relevant_licence_data[0][0],
         relevant_licence_data[0][1], relevant_licence_data[0][2],
         relevant_licence_data[0][3])

   widget.delete(0,"end")
```

```
main_frame = tk.Tk()

main_frame.title("Template Recognition")

main_frame.geometry("550x220+30+30")

# main_frame.iconbitmap(default='icon.ico')

root = tk.Frame(main_frame,bg = "cyan")

root.place(x=0,y=0,width = 550, height = 220)

text_field = tk.Text(root)

text_field.place(x = 10, y = 10, width = 530, height = 200)

text_field.insert(0.0,print_adhaar_data+print_licence_data)
```

5.4 Result Analysis

The Result was analyzed while development of the system and it was made pretty accurate. The System worked reliably. The system will be tested further using several test cases and scenarios.

CHAPTER 6

TESTING

6.1 Overview of Testing

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

6.2 Types of Testing

6.2.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.2.2 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.2.3 Functional testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.
Systems/Procedures	: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

6.3 Unit Testing:-

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.
Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.
- Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

6.4 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

6.5 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

6.6 Test Cases

Test cases can be divided in to two types. First one is Positive test cases and second one is negative test cases. In positive test cases are conducted by the developer intention is to get the output. In negative test cases are conducted by the developer intention is to don’t get the output.

6.7 Black Box Testing

Black Box Testing is a software testing method in which the functionalities of software applications are tested without having knowledge of internal code structure, implementation details and internal paths. Black Box Testing mainly focuses on input and output of software applications and it is entirely based on software requirements and specifications. It is also known as Behavioral Testing.

6.8 White Box Testing

White Box Testing is software testing technique in which internal structure, design and coding of software are tested to verify flow of input-output and to improve design, usability and security. In white box testing, code is visible to testers so it is also called Clear box testing, open box testing, transparent box testing, Code-based testing and Glass box testing

6.9 Design of Test Cases and Scenarios

	TEST CASES/ SCENARIOS	EXPECTED	ACTUAL	VALIDATED
1	Selecting an Image	The System should be able to load various image formats and show them.	The System was able to load various image formats and show them.	YES
2	Get Details from Aadhar Card	The System should be able to fetch and parse the details from the given Aadhar Card Image	The System was able to fetch and parse the details from the given Aadhar Card Image	YES
3	Save to Database	The Details fetched and Parsed from the	The Details fetched and Parsed from	YES

		Aadhar Card should be save in the database	the Aadhar Card was save in the database	
4	Search the Database	The User should be able to fetch and search the Student details from Database.	The User wasable to fetch and search the Student details from Database.	YES

CHAPTER 7

SCREENTHOTS

7.4 Interface Screenshot

7.7.1 Upload Page

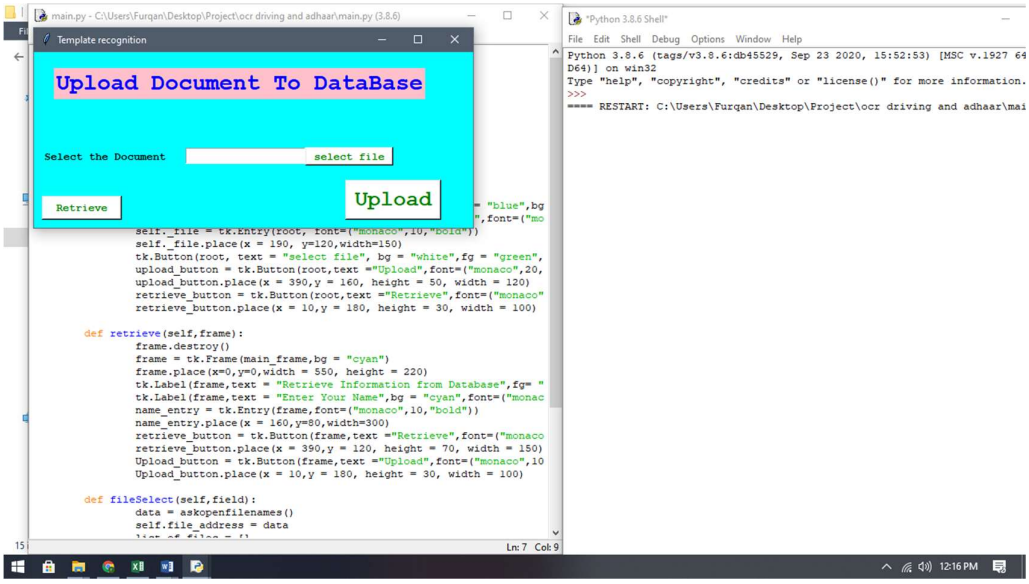

Figure 7.1 Upload Page

Above Image show the Upload Data page interface of Project

7.7.2 Retrieve Page

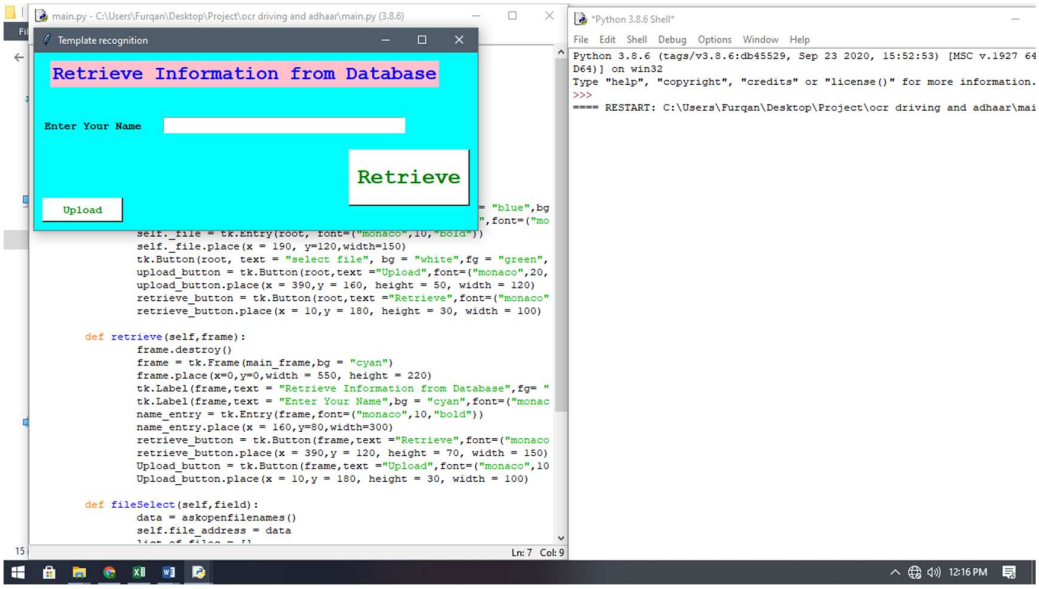

Figure 7.2 Retrieve Page

Above Image show the Retrieve Data page interface of Project

CHAPTER 8

CONCLUSION

8.1 Project Conclusion

The literature review acknowledges the work of previous researchers and in doing so assures readers that their work is well-conceived. It is assumed that by referring to previous works in the research field, the author has read, evaluated and incorporated this work into the Proposed Project. The literature review creates a "landscape" for the reader, allowing him to fully understand the development of the field. This overview tells the reader that the author has in fact absorbed all (or most) of the previous important work in this field in his/her research and now have gathered enough knowledge to start the analysis of the own proposed Project.

The System for Automating the Process of Student Registration was Designed and Analyzed in real conditions with scanned Images of Aadhar Cards. The System was able to pass the designed Test Cases.

8.2 Limitation of Project:

The System needs a clear image of the Aadhar Card of the Student to work, Although this limitation is very easy to solve by using any general Scanners.

8.3 Future Enhancements:

The system can be upgraded in future by adding following Features:

- By adding a feature to automatically generate ID card based on the details.
- We can add features like OTP authentication to make system secure.

REFERENCES

[1] Rafi, Ali, Faraz, Athaul, “OCR Engine to extract Food-items and Prices from Receipts Images via Pattern matching and heuristics approach”, SMIU, 1st International Conference on computing and related technologies, 2017 [Souvik Das “The Development of a Microcontroller Based Low Cost Heart Rate Counter for Health Care Systems” International Journal of Engineering Trends and Technology-Volume4Issue2- 2013.

[2] Chaki, Nabendu, Soharab Hossain Shaikh, and Khalid Saeed. "A comprehensive survey on image binarization techniques." In Exploring Image Binarization Techniques, pp. 5-15. Springer India, 2014.

[3] Zhang, Mi, Anand Joshi, Ritesh Kadmawala, Karthik Dantu, Sameera Poduri, and Gaurav S. Sukhatme. "OCRdroid: A Framework to Digitize Text Using Mobile Phones." In MobiCASE, pp. 273-292. 2009.

[4] Brisinello, Matteo, Ratko Grbić, Matija Pul, and Tihomir Anđelić. "Improving Optical Character Recognition Performance for Low Quality Images." In 59th International Symposium ELMAR-2017. 2017.

[5] ZHAO, Yan, Yue CHEN, and Shi-gang WANG. "Corrected fast SIFT image stitching method by combining projection error." Optics and Precision Engineering 6 (2017): 029.