Document Text Extraction using Optical Character Recognition (OCR)

Teena Shaik – AP24122040002
Piyush Keshari – AP24122040004
Ridhamkumar Thumar – AP24122040014

1. Abstract

Optical Character Recognition (OCR) is an incipient technology that allows to meets the effort and render the captured text searchable, and editable by converting a scan of document into machine comprehensible typescript. This project is about OCR implementation in Python using TesseractOCR and OpenCV. In this project we are extracting the text from the given sets of documents along with the required image. It is a web application which can be accessed anywhere and the extracted text will be available in proper format. Couple of sample documents Like, aadhar card, pan card, passport and PDFs are taken and text extraction is performed. Trial of adding voice assistant for specially challenged people to the web application is also under process and testing for the same is yet to be done.

2. Methodology description

The overall project is divided into several parts. The first part consists of the backend part which consists of several modules like pytesseract, openCV and virtual environment. For the database we are using postgresql for storing the data. The backend part is the core part in which input image and PDFs will be passed as an input and the resultant output will be formatted as a dictionary.

For the second part we are using a python based web library **Flask** which will act as a bridge between our backend and the frontend. The User Interface (UI) is kept very simple. As stated earlier we are also trying to inculcate a virtual voice assistant for specially challenged people. For that different mechanisms are being tested.

3. Dataset Statistics

The proposed database will store different sets of data with following attributes**,**
The extracted image data will be flattened and stored in the jsonb format of postgresql, the extracted textual data will be stored in the database only with the consent from the user. If the user wants to store the data then the data will be stored in json format.

4. **Data flow architecture from frontend to backend.**

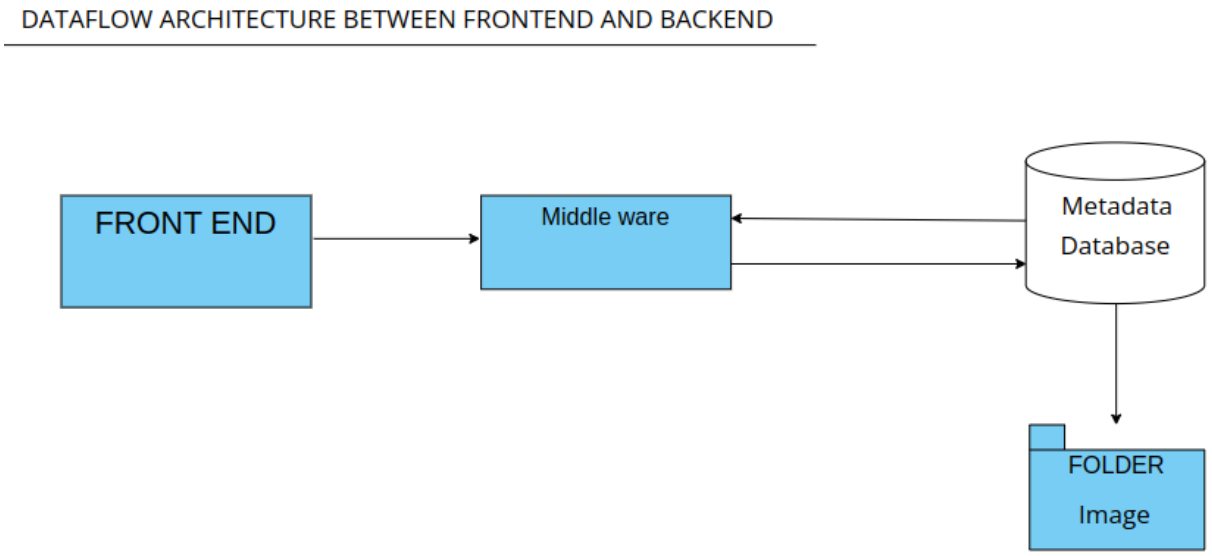


Fig1. Flow of data between frontend, middleware and database.

The data that we are dealing with majorly contains images of different types of data along with other types of textual data. The uploaded image via the frontend will be passed to the middleware and the data if consented by the user will be stored to the database. The database contains the metadata for all the data that the user wants to store. In metadata the path of the image data is stored in encrypted format and the data can be retrieved by using the appropriate decryption function. Below is the given ER diagram of the database table that will contain the data.

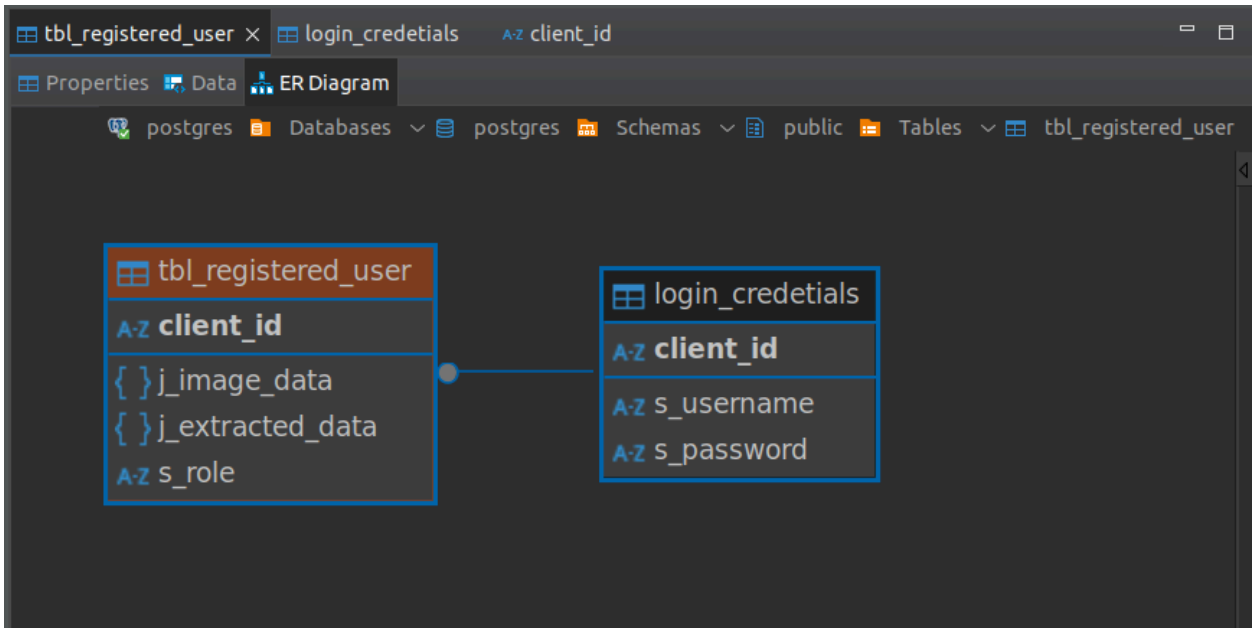


Fig 2. E-R Diagram of the proposed database table

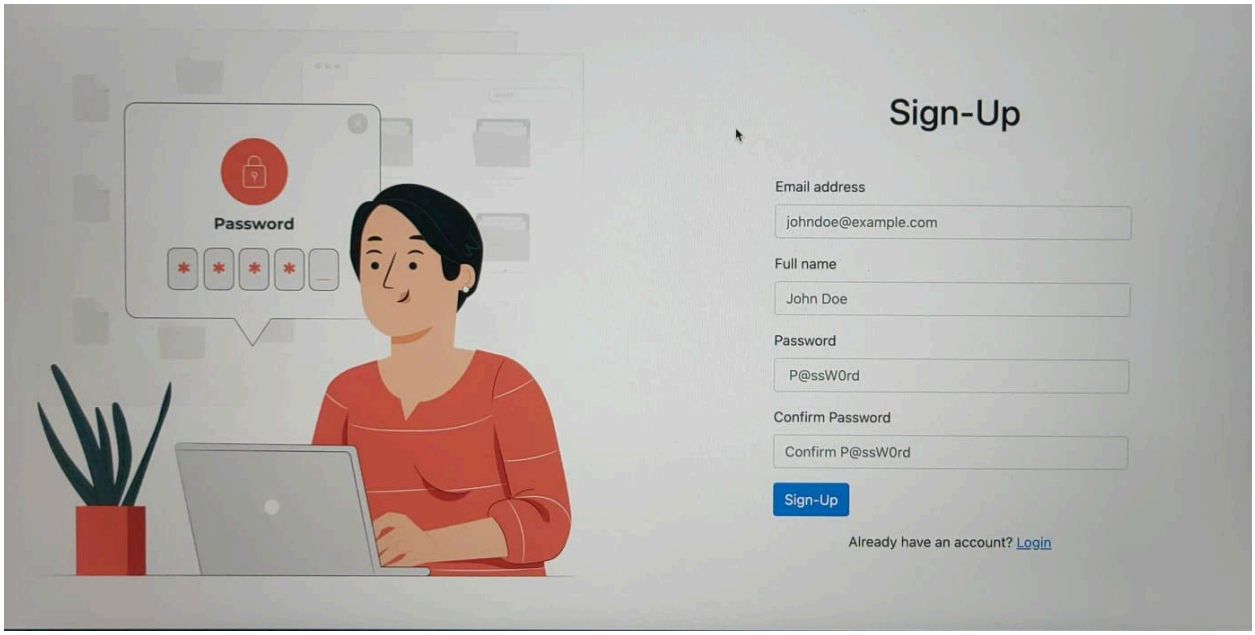
Here the client_id is the foreign key and is referenced by the login_credentials table. BTree indexing is applied on the client_id column of the login_credetials.

5. Results Obtained

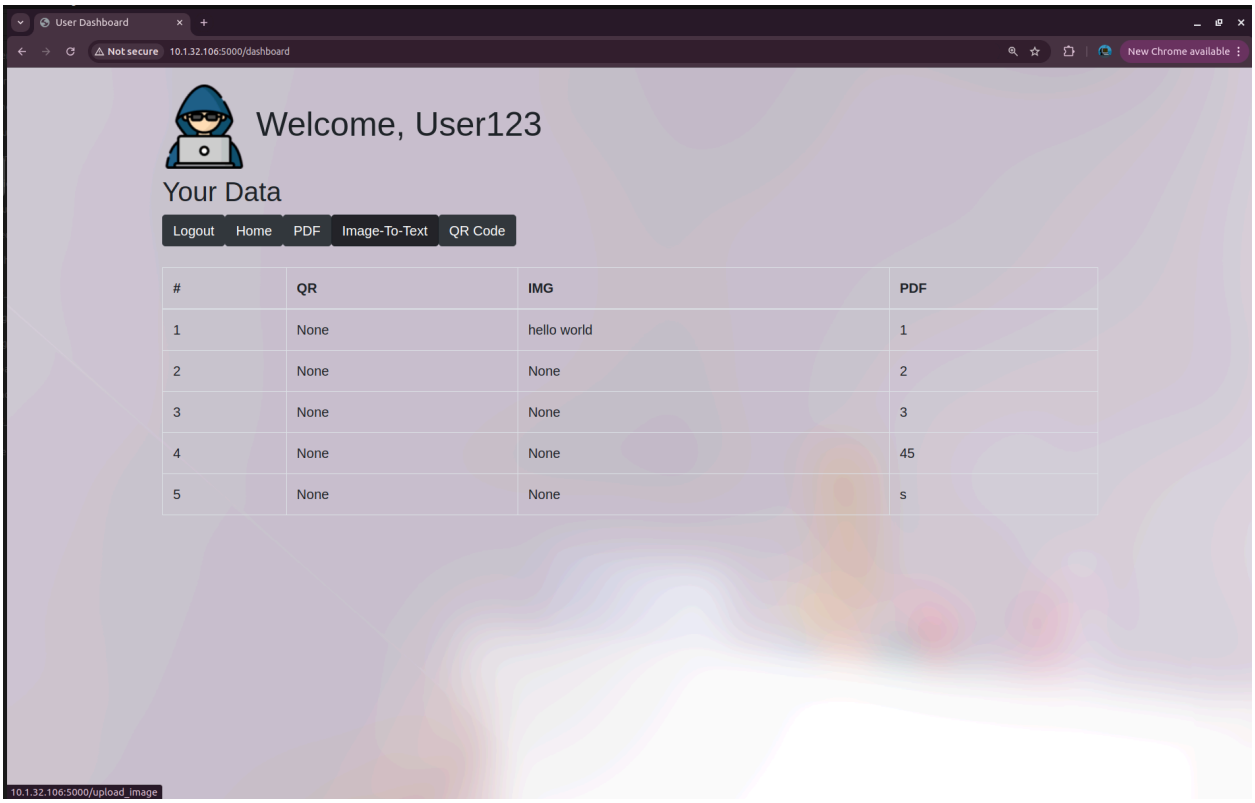
The initial prototype of the project is tested and it is working ok. Further some more fine tuning needs to be done before deploying the site.

Different parts of the site are given below:

I. *Sign up / Login page*

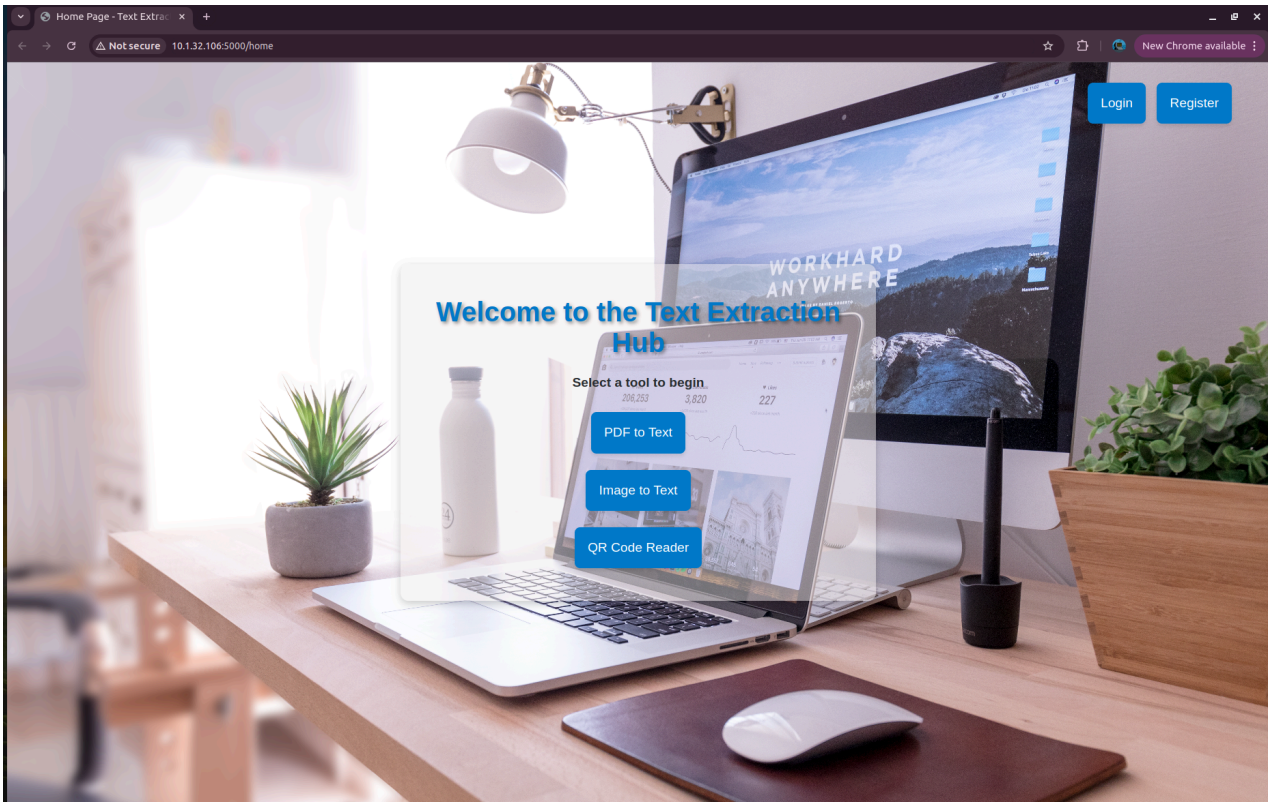


II. Dashboard of user logged in



Main home page

The name of the login button changes to the user's name after logging in successfully.

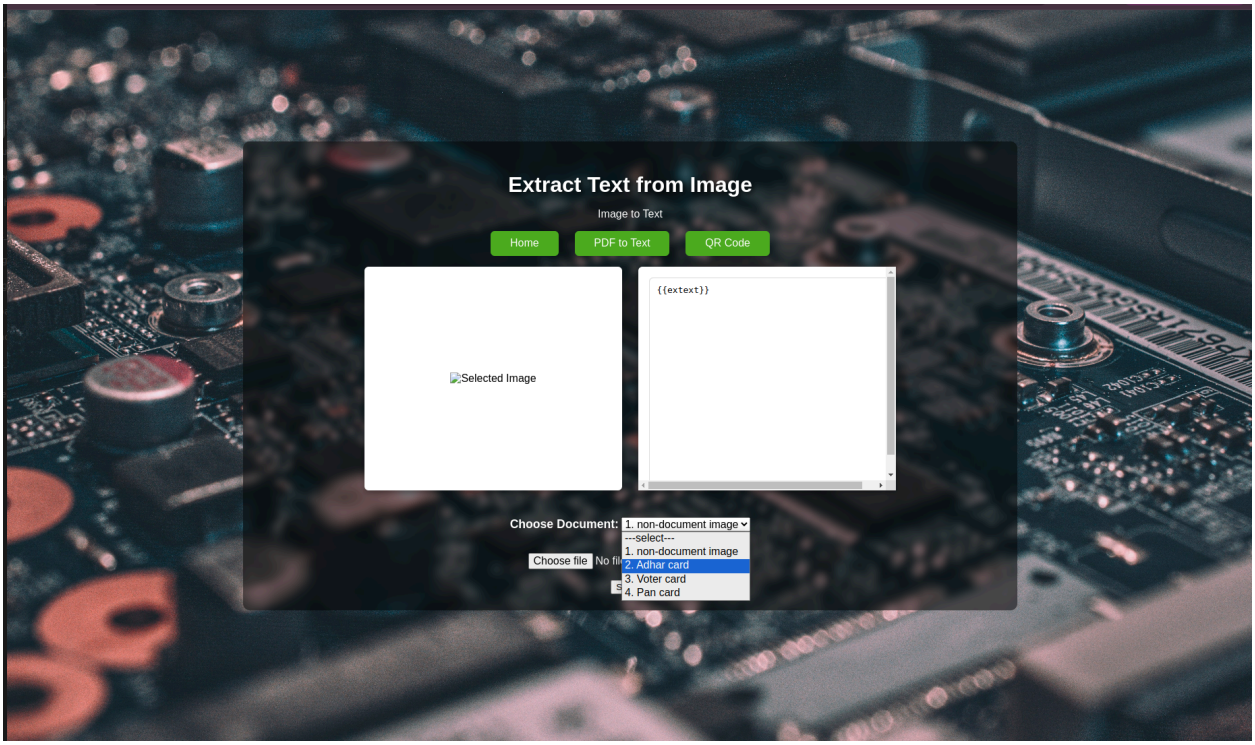


III. *PDF text extraction.*

This page is responsible for extracting the text from a pdf file. There is also an option for extracting the text from the pdf by specifying the page number.

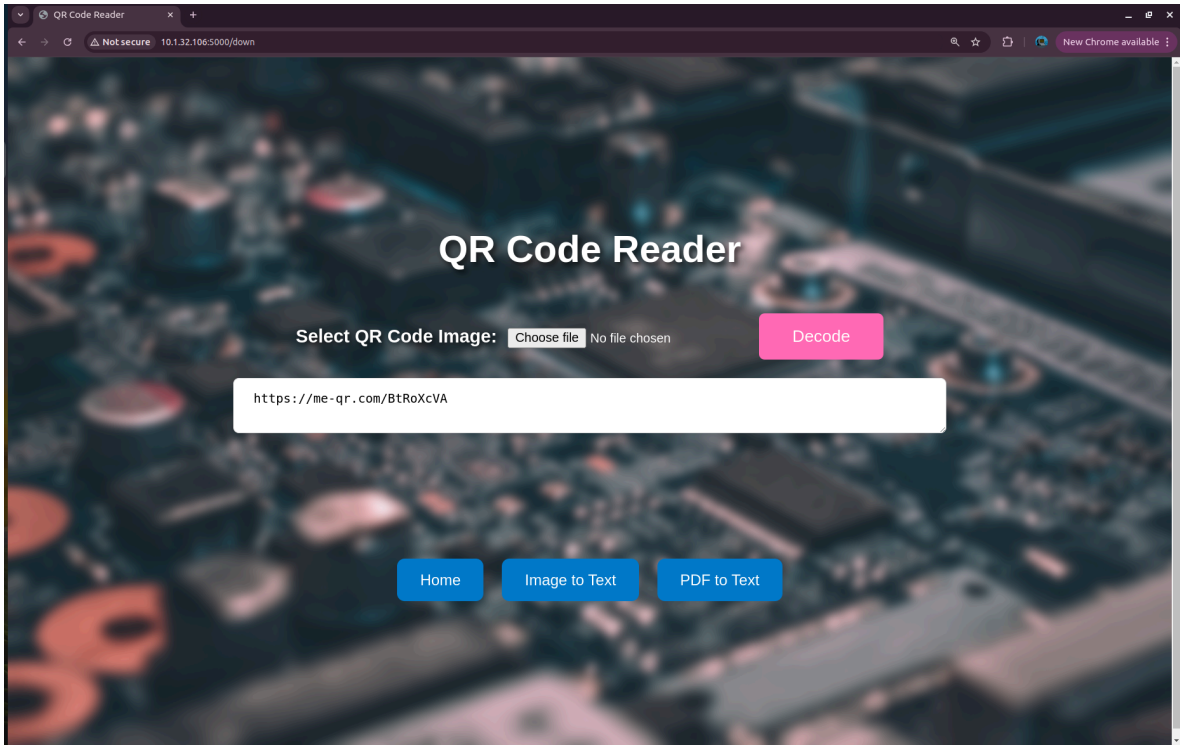
IV. *Text extraction from different types of image documents*.

This page is capable of extracting text from different types of documents as per the user requirements.



V. *QR code reader*

There is also a provision for the user to input any QR code image to extract the text out of it



6. File Structure

```
Doc_text_extract
├── templates
│   ├── homepage.html
│   ├── login.html
│   ├── qcode.html
│   ├── imgtext.html
│   ├── homepage_user.html
│   ├── app.html
│   ├── pdftext.html
│   ├── dashboard.html
│   └── register.html
├── README.md
├── static
│   ├── QRCODE
│   │   ├── test_qr_2.png
│   │   └── test_qr.png
│   ├── pdfiles
│   │   ├── Assignment-1_report.pdf
│   │   └── Cancer detection.pdf
│   ├── home_background.png
│   ├── adhar
│   │   └── test_image.jpg
│   ├── pan
│   └── hacker.png
├── app.py
└── processing_modules
    ├── pdfextract.py
    ├── adharcard_long.py
    ├── QRreader.py
    ├── config_generator.py
    ├── image_to_text.py
    ├── databse.py
    ├── encrypt_decrypt_data.py
    ├── query.py
    └── config.ini
```

7. Future Scope

Since this is the beta version there are lots of things that can be added. Due to time constraints the voice helper is not yet enabled on which the work is still going. There are a lot of things that can be integrated into this website.

8. Conclusion

The project was aimed not only in extracting text out of any government document but also it played a multirole in extracting text out of PDF files and QR codes also. The simplicity in the User interface(UI) will also help in getting along with the specially challenged people. That part is still under development.

9. Reference

I. flask Documentation source: https://flask.palletsprojects.com/en/stable/
II. Image source: https://images.unsplash.com/photo-1518770660439-4636190af475
III. Icon source: https://www.flaticon.com/free-icons/user/2