Django OCR Application: OCR and Data Extraction

Name: **Parth Narendra Barse** Email: parth.barse.work@gmail.com Phone: **8793015610**

This Django application leverages Optical Character Recognition (OCR) to extract text from uploaded images, whether from file uploads or live camera captures. The extracted text is then displayed on the web interface and can be downloaded as a text file or PDF. All processed text data and related metadata are stored in a MySQL database for future reference.

This Project is Created for **Assignment 1 : OCR and Data Extraction**

(Add. : 20, Sahajanand-1, near Gandhi Bhavan, Kothrud, PUNE- 411 038 Phone: +91 98900 33285 | E-Mail: sandeep@automationteknix.com | Web: www.automationteknix.com)

Objective: Develop a Django application that uses OCR to extract text from uploaded images and display the results.

Tasks:

1. Set up Django Project: Create a new Django project and set up the necessary apps.

2. OCR Implementation:

a) Integrate an OCR library like Tesseract with Python (using pytesseract) to extract text from images.

b) Create a form in the Django app to upload images

c) Also with your laptop's webcam, you can capture image and process it.

d) Both uploading image and webcam options should be in your form.

e) Process the uploaded images using the OCR library and extract text.

3. Display Results:

a) Show the extracted text on a new page or in the same view.

b) Allow users to download the extracted text as a text file or a PDF.

4. Data Storage: Store the extracted text and metadata in a database for future reference. Use MySQL (using xampp server), ssms (SQL server Management studio), SQLite.

Table of Contents

Introduction

This Django application was developed to provide an easy way to extract text from images using OCR. Users can either upload an image or capture one using their webcam. The application processes the image to extract text and allows users to download the extracted text in TXT or PDF format. Additionally, all text data along with metadata is stored in a MySQL database and can be viewed on a history page.

Installation and Setup

Prerequisites

- **XAMPP Server**: Ensure that XAMPP server is installed and running for Apache and MySQL on ports 80 and 3306, respectively.
- **Tesseract-OCR**: The application requires the Tesseract OCR engine. Follow the Tesseract installation guide to install it. Use the default installation path.

Installation Steps

1) **Clone the Repository**: Clone the application from the GitHub repository.

   git clone
   https://github.com/ParthBarse/django-assesment-task.git

   cd django-assesment-task

2) **Install Dependencies**: Install the required Python packages using pip.

   pip install -r requirements.txt

3) **Database Setup**: Make sure your MySQL database is running and create a database named ocr_data using the XAMPP MySQL admin interface (phpMyAdmin).

- **Database Credentials**:
  - Username: root (default)
  - Password: (leave it empty)
  - Database Name: ocr_data

4) **Run Migrations**: Create the necessary database tables by running Django migrations.

   python manage.py makemigrations
   python manage.py migrate

5) **Run the Server**: Start the Django development server.

   python manage.py runserver

6) **Access the Application**: Open your web browser and navigate to http://localhost:8000 to access the application.

Usage

Home Page

- **Upload Image**: Choose an image file from your computer and upload it to extract text.
- **Live Camera Capture**: Use your webcam to capture an image and process it for text extraction.

History Page

- View previously extracted text and metadata stored in the MySQL database.

Download Options

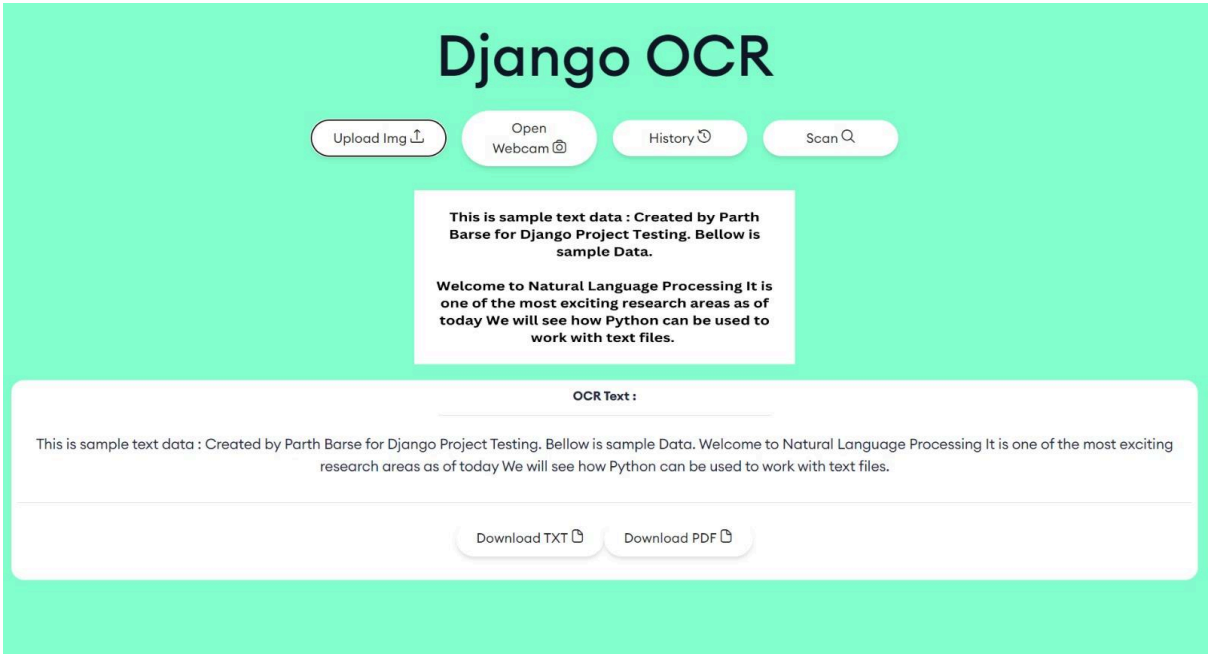
- After extracting the text, you can download it as a .txt or .pdf file.

Features

- **Image Upload**: Upload images from your local file system for OCR processing.
- **Live Camera Capture**: Capture images directly from your webcam for OCR processing.
- **Image Retake**: Retake the image if the initial capture was not satisfactory.
- **Text Download**: Download the extracted text in either TXT or PDF format.
- **Data Storage**: All OCR text and associated metadata are stored in a MySQL database and can be accessed via the history page.
- **Efficient Memory Usage**: The processed images are not saved to disk, ensuring minimal memory usage.

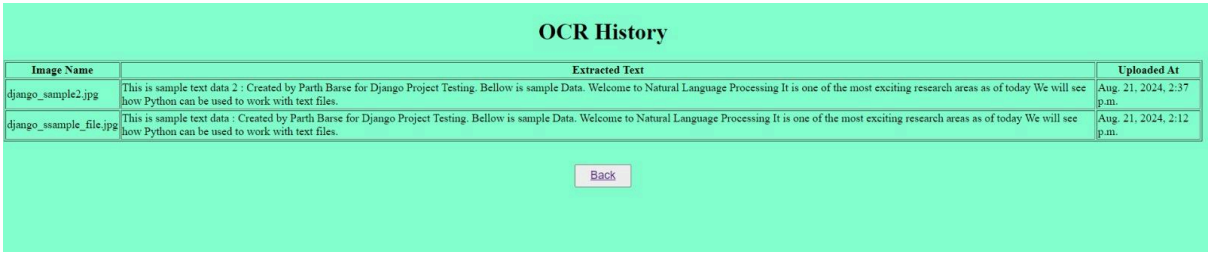
Screenshots

Home Page (Upload Image)



Home Page (Live Camera Capture)

History Page (MySQL Database Stored Data)



OCR History

Image Name	Extracted Text	Uploaded At
django_sample2.jpg	This is sample text data 2 : Created by Parth Barse for Django Project Testing. Bellow is sample Data. Welcome to Natural Language Processing It is one of the most exciting research areas as of today We will see how Python can be used to work with text files.	Aug. 21, 2024, 2:37 p.m.
django_ssample_file.jpg	This is sample text data : Created by Parth Barse for Django Project Testing. Bellow is sample Data. Welcome to Natural Language Processing It is one of the most exciting research areas as of today We will see how Python can be used to work with text files.	Aug. 21, 2024, 2:12 p.m.

Back

Frameworks and Libraries Used

- **Django**: A high-level Python web framework that encourages rapid development and clean, pragmatic design.
- **Tailwind CSS**: A utility-first CSS framework for rapid UI development.
- **pytesseract**: A Python wrapper for Google’s Tesseract-OCR Engine, used for text extraction from images.
- **Fpdf**: A Python library to generate PDF files.

Troubleshooting

- **Database Connection Issues**: Ensure that MySQL is running on the default port and that the database credentials in the Django settings file are correct.
- **Tesseract-OCR Installation**: If the OCR feature isn't working, verify that Tesseract-OCR is correctly installed and accessible from the system path.