n Image to Text Converter using Django and Tess

Professional Office Demo

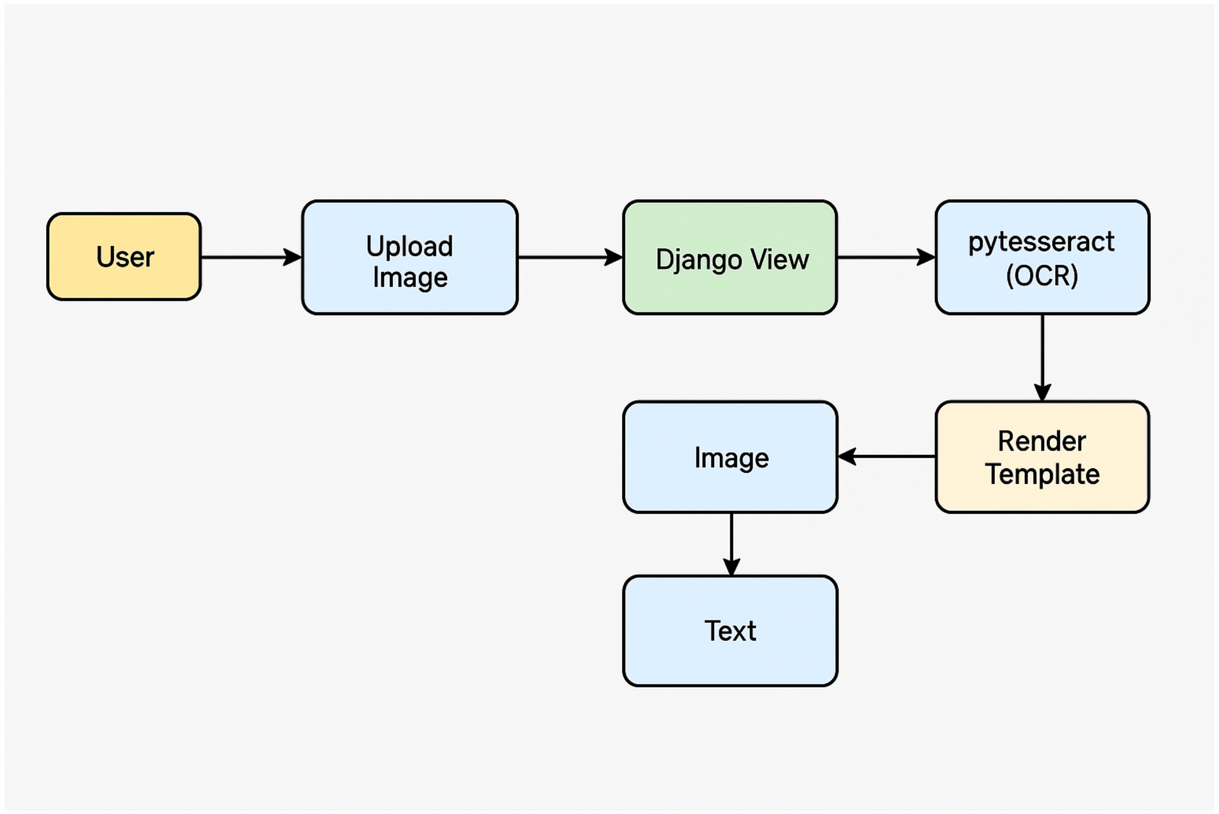
October 07, 2025

Prepared by: Sanjay.V

Agenda

1. Project Overview
2. Architecture & Flow
3. Tech Stack & Libraries
4. Installation & Setup
5. Demo & UI
6. Code Highlights
7. Next Steps & Q&A

Architecture Overview



Data Flow

1) User uploads image via web form.

2) Django view saves file to static/uploads and opens it with Pillow.

3) pytesseract.image_to_string extracts text from the image.

4) View stores the extracted text in session (or database) and renders template.

5) Template displays image (left) and text (right). Users can copy text or download as PDF.

Installation & Setup

1) Virtual Environment

```
python -m venv venv
Windows: venv\Scripts\activate   macOS/Linux: source venv/bin/activate
```

2) Install Python Packages

```
pip install django pillow pytesseract reportlab
```

3) Install Tesseract OCR Engine

Windows: download installer (UB-Mannheim or GitHub releases) and add to PATH

```
macOS: brew install tesseract    Ubuntu: sudo apt install tesseract-ocr libtesseract-dev
```

4) Django Setup

```
python manage.py makemigrations
python manage.py migrate
python manage.py runserver
```

Code Highlights

views.py (core logic)

```
if request.method == 'POST' and request.FILES.get('image'):
    uploaded_file = request.FILES['image']
    path = os.path.join('ocr_app', 'static', 'uploads', uploaded_file.name)
    with open(path, 'wb+') as dest:
        for chunk in uploaded_file.chunks():
            dest.write(chunk)
    img = Image.open(path)
    extracted_text = pytesseract.image_to_string(img)
    request.session['extracted_text'] = extracted_text
```

download_pdf view (ReportLab)

```
buffer = BytesIO()
p = canvas.Canvas(buffer)
for line in text.split('\n'):
    p.drawString(50, y, line)
    y -= 15
p.save()
return FileResponse(buffer, as_attachment=True, filename='Extracted_Text.pdf')
```

Core Logic Explanation

1. File Upload: User uploads an image through a Django form, received in request.FILES.
2. File Saving: The uploaded image is saved under 'static/uploads/' using binary mode to ensure proper image data handling.
3. OCR Extraction: Pillow opens the image, and pytesseract.image_to_string() extracts readable text.
4. Session Storage: Extracted text is stored in Django session for later use (display & PDF export).
5. Rendering: The HTML template displays the uploaded image on the left and extracted text on the right.
6. PDF Generation: Using ReportLab, text is written line by line into a dynamic PDF for download.

Flow Summary: User → Upload → Django View → Pillow → pytesseract → Session → Template → ReportLab → PDF.

Key Libraries Used:

• Django – Web framework and session management

• Pillow – Image opening and preprocessing

• pytesseract – OCR engine wrapper for Tesseract

• ReportLab – PDF generation and download

Next Steps & Contact

- Save upload history to database with timestamp and user metadata.
- Improve accuracy with TrOCR or fine-tune models for specific handwriting.
- Add language selector and multi-script support (e.g., Tamil, Hindi).
- Provide REST API endpoints and Dockerize for deployment.

Questions? Contact: Sanjay.V