main.py

```
from flask import Flask, request, render_template, send_file
import pytesseract
from PIL import Image
import pdfplumber
import os
from utils import extract_entities, fill_form

app = Flask(__name__)

UPLOAD_FOLDER = "uploads"
OUTPUT_FOLDER = "output"
os.makedirs(UPLOAD_FOLDER, exist_ok=True)
os.makedirs(OUTPUT_FOLDER, exist_ok=True)

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/upload', methods=['POST'])
def upload_file():
    file = request.files['document']
    if file.filename == '':
        return "No selected file", 400
    filepath = os.path.join(UPLOAD_FOLDER, file.filename)
    file.save(filepath)

    text = ""
    if file.filename.lower().endswith('.pdf'):
        with pdfplumber.open(filepath) as pdf:
            for page in pdf.pages:
                text += page.extract_text() or ""
    else:
        img = Image.open(filepath)
        text = pytesseract.image_to_string(img)

    entities = extract_entities(text)

    filled_form_path = fill_form(entities, OUTPUT_FOLDER)

    return send_file(filled_form_path, as_attachment=True)

if __name__ == '__main__':
    app.run(debug=True)
```

second_file.py

```
# AI-Powered Form Filling Assistant

## Overview
A simple Flask web app that extracts key information (Name, DOB, Address,
Aadhaar, PAN) from uploaded scanned documents (PDF/images) and produces a
pre-filled Word document.

## Requirements
- Python 3.9+ recommended
- Tesseract OCR installed on the system and accessible in PATH.

## Install Tesseract (important)
```

- **Ubuntu / Debian**: `sudo apt-get install tesseract-ocr`
- **Windows**: Download installer from https://github.com/tesseract-ocr/tesseract and install. Add tesseract to PATH.

Setup
```bash
python -m venv venv
source venv/bin/activate   # Linux/Mac
venv\Scripts\activate    # Windows (PowerShell)
pip install -r requirements.txt
python -m spacy download en_core_web_sm
python main.py
```

Open http://127.0.0.1:5000/ in your browser.

Notes & Next Steps
- This is a prototype: entity extraction uses heuristics and regex. For production, use a trained NER model and more robust OCR (Google Vision, AWS Textract).
- To handle Indian languages, integrate Indic OCR or translation services.
- To support handwritten text, use advanced OCR like Google Vision or fine-tuned models.

```
Flask==3.0.0
pytesseract==0.3.10
Pillow==10.0.0
pdfplumber==0.10.3
spacy==3.7.2
python-docx==1.1.0
```

```
third_file.py
import re
from docx import Document
import os

def extract_entities(text):
    entities = {}

    # Aadhaar number pattern (with or without spaces)
    aadhaar = re.findall(r"\b\d{4}\s\d{4}\s\d{4}\b", text) or re.findall(r"\b\d{12}\b", text)
    if aadhaar:
        entities['Aadhaar'] = aadhaar[0]

    # PAN card pattern
    pan = re.findall(r"[A-Z]{5}[0-9]{4}[A-Z]{1}", text)
    if pan:
        entities['PAN'] = pan[0]

    # Date of Birth pattern (dd/mm/yyyy or dd-mm-yyyy)
    dob = re.findall(r"\b\d{2}[/-]\d{2}[/-]\d{4}\b", text)
    if dob:
        entities['DOB'] = dob[0]

    # Name (simple heuristic: first non-empty line with two or more words)
```

```
    lines = [line.strip() for line in text.splitlines() if line.strip()]
    for line in lines:
        if len(line.split()) >= 2 and not any(char.isdigit() for char in
line):
            entities['Name'] = line
            break

    # Address detection (heuristic)
    address_lines = [line for line in lines if any(word in line for word
in ['Road', 'Street', 'Nagar', 'Colony', 'City', 'District', 'Pin',
'PIN', 'Pincode'])]
    if address_lines:
        entities['Address'] = ', '.join(address_lines[:3])

    return entities

def fill_form(entities, output_folder):
    doc = Document()
    doc.add_heading('Government Service Form (Auto-Filled)', 0)

    if not entities:
        doc.add_paragraph('No entities were detected in the uploaded
document.')
    else:
        for key, value in entities.items():
            doc.add_paragraph(f"{key}: {value}")

    os.makedirs(output_folder, exist_ok=True)
    filepath = os.path.join(output_folder, "filled_form.docx")
    doc.save(filepath)
    return filepath
```