



Demonstrasi Pipeline OCR

Pengarsipan Digital Dokumen Kepemilikan Tanah

Disusun oleh Julian Emir Prawira

Pendahuluan (1) – Outline

- Nama Program Kerja:
 - Program Kerja 1: Strukturisasi Data dan Arsip Digital Dokumen Pemilikan Tanah
 - Program Kerja 2: Alat Pencarian Dokumen Kepemilikan Tanah
- Bagian dari KKN PPM UGM 2024/2025 (Periode 4)
- Lokasi: Kelurahan Randuacir
- Tujuan: Mengembangkan alat pengambilan dokumen berbasis OCR untuk mendigitalkan catatan tanah.

Pendahuluan (2) – Mengapa Proyek Ini?

- Masalah Arsip Fisik Saat Ini:
 - Rawan terhadap kerusakan (usia, kelembaban, kondisi lingkungan)
 - Sulit mencari informasi tertentu
 - Pencatatan manual tidak efisien
- Solusi OCR (*Optical Character Recognition*):
 - Mengubah dokumen fisik menjadi data digital yang terstruktur
 - Memungkinkan pencarian dan pengambilan cepat
 - Melindungi dokumen dari kerusakan fisik

Pendahuluan (3) – Tahapan Implementasi

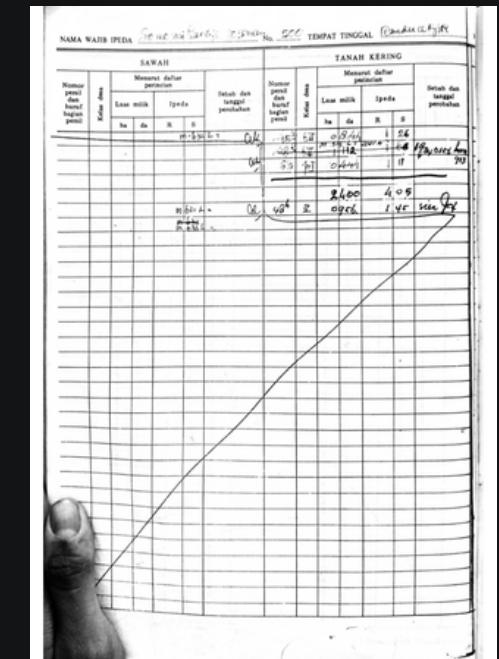
- Tahap Pertama: OCR diuji pada satu sampel dokumen
- Tahap Kedua: Peningkatan skala untuk memproses ratusan catatan kepemilikan tanah
- Tantangan
 - Masalah kualitas dokumen (teks memudar, catatan tulisan tangan, sobekan, noda)
 - Efisiensi penyimpanan & pemrosesan data skala besar
 - Peningkatan akurasi melalui praproses & pengoptimalan

Gambaran Umum Alur Kerja (1) – Ikhtisar Alur Kerja

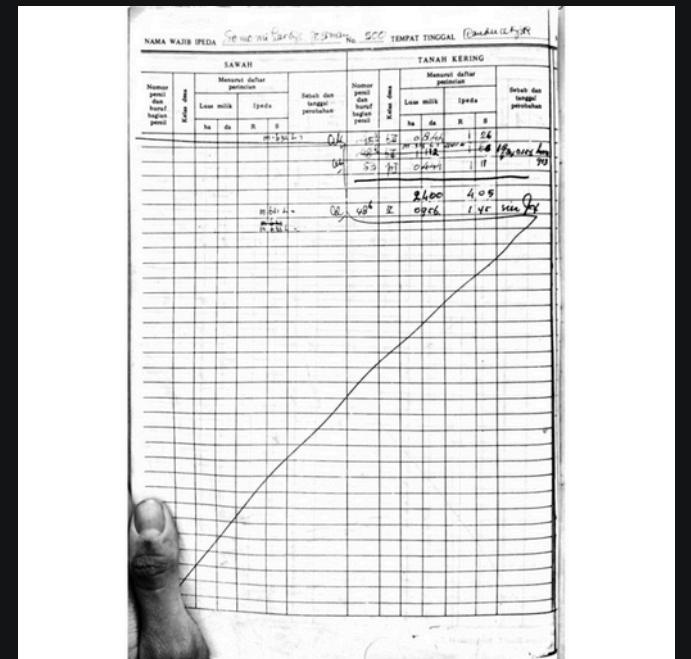
- Sasaran: Mengonversi dokumen kepemilikan tanah fisik menjadi data digital terstruktur
- Langkah-langkah:
 - Mengonversi PDF ke PNG → Pemrosesan lebih mudah
 - Memroses Gambar → Meningkatkan kualitas untuk OCR
 - Mengekstrak Teks menggunakan OCR → Mengonversi data gambar ke teks
 - Menyimpan Hasil dalam CSV → Diatur untuk pencarian & pengambilan

Gambaran Umum Alur Kerja (2) – Detail Praproses

- Mengapa praproses sangat penting?
 - Pemindaian berkualitas buruk memengaruhi akurasi OCR
- Langkah-langkah Praproses:
 - Konversi ke *Grayscale*
 - Tingkatkan Kontras (metode CLAHE)
 - Hapus Noise (*Median Blurring*)
 - Normalisasi Dimensi Gambar



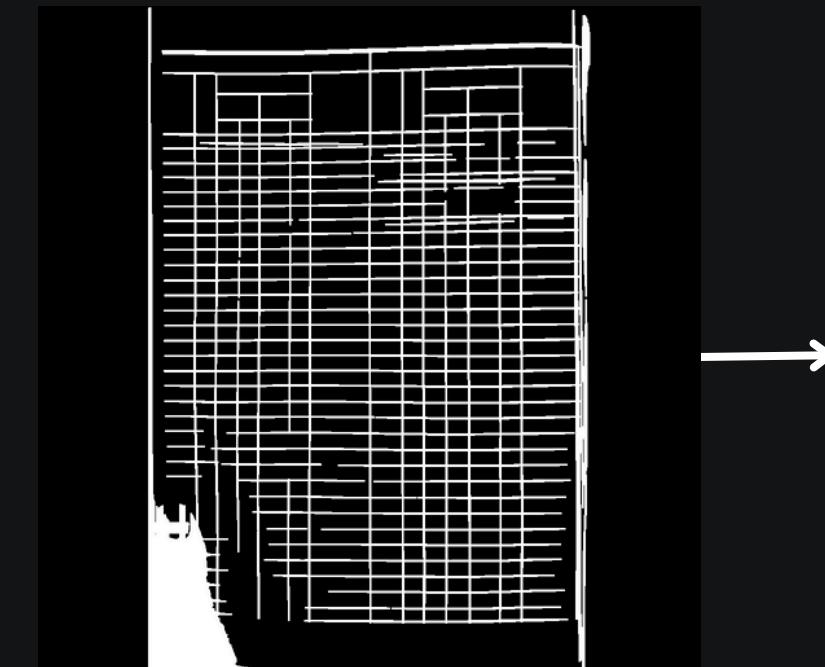
PDF halaman 500 (dari sumber)



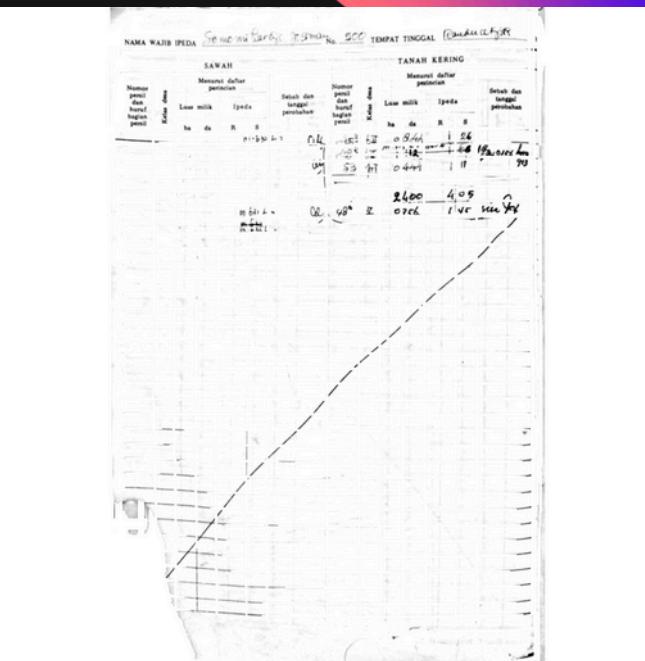
Hasil halaman 500 setelah praproses

Gambaran Umum Alur Kerja (3) – Menjalankan OCR

- Pemrosesan OCR:
 - Mendeteksi teks dalam gambar
 - Menghapus elemen yang tidak diperlukan (tabel, batas)
- File Keluaran:
 - Gambar yang Diproses
 - File Teks Mentah & Bersih dalam bentuk .txt
 - File CSV Final
- Pertimbangan Ukuran File:
 - Memproses ratusan dokumen dapat memakan ruang hingga 6GB, disarankan untuk memiliki setidaknya 10GB ruang disk SSD yang tersedia



Tabel yang diekstraksi dari halaman 500



Hasil dari ekstraksi tabel

Page	OCR Output
halaman_1	NAMA WAJIB IPEDA RING akan TANAH KE Menara daftar raindan Nomor eincan
halaman_10	SC raat Tmocal bedi TANAH KERING Manure atei HI sabe dan Cy Lone mitt pane
halaman_11	EE Barclarm Ber Rect 74 NAMA WANB IPEDA DO MO MAOH vie 5H. Tempat Tinco
halaman_12	rewpat TingoaL ff TANAH KERING Menurat daftar perincian M198
halaman_13	rr r re SAWAK TANAH KERING Menurst dafter ae Menurvt dafter Nemer penocen
halaman_14	No 37 veMpat TINGGAL Lui er Se MEO TANAH KERING Menaret daara a Ipods en
halaman_15	NAMA wan treDn COMBISG.Wit Cel Kase, A1E.. TEMPAT TINGOAL AAMIHOAR SA
halaman_16	AP No wc Sld. TEMPAT TINGGAL CMEC TANAH KERING a Menure dafter Menurut
halaman_17	NAMA WAJIB IPEDA aseey 7 Soin Ne SUG, rena incon, DAAMOCUR. TANAH KERI
halaman_18	Mw tuner Naan WAIT TPEDA SO ID nes NO BID. TEMPAT TINGGAL SAWAH TANAH
halaman_19	i Sidon NAMA WAJIB IPEDA OCEMCY 10. Dyaealn ne S20 rewrat nincca, RE. SAWA
halaman_2	awa waste IPEDA No. 7D TEMPAT TINGGAL SAWAM TANAH KERING Menarei dafr
halaman_20	. aie 4 No 4 rewPaT TINGGAL TANAH KERING meiaeae sh Nomor ae is Seba dan c
halaman_21	oe wun wane wean Sersege ttfe, Mr nq 522. rowrar conn, fla TANAH KERING s m

20 Halaman Pertama pada File output CSV

Instalasi dan Persiapan (1) – Persyaratan Sistem

- Telah di uji-coba pada:
 - OS: Windows 10
 - CPU: Intel i7-10750H @ 2.60GHz (12 CPU threads)
 - RAM: 16 GB DDR4 dual-channel
 - GPU: Nvidia GTX 1650 Ti
 - Penyimpanan: SSD 500 GB
- Spesifikasi yang direkomendasikan:
 - OS: Windows 10 dan keatas
 - CPU: Intel Core i5 (Generasi ke-8) atau lebih tinggi
 - RAM: minimal 8 GB
 - Penyimpanan: SSD dengan ruang kosong minimal 10 GB
 - GPU: Tidak diperlukan (pipeline berjalan pada CPU)

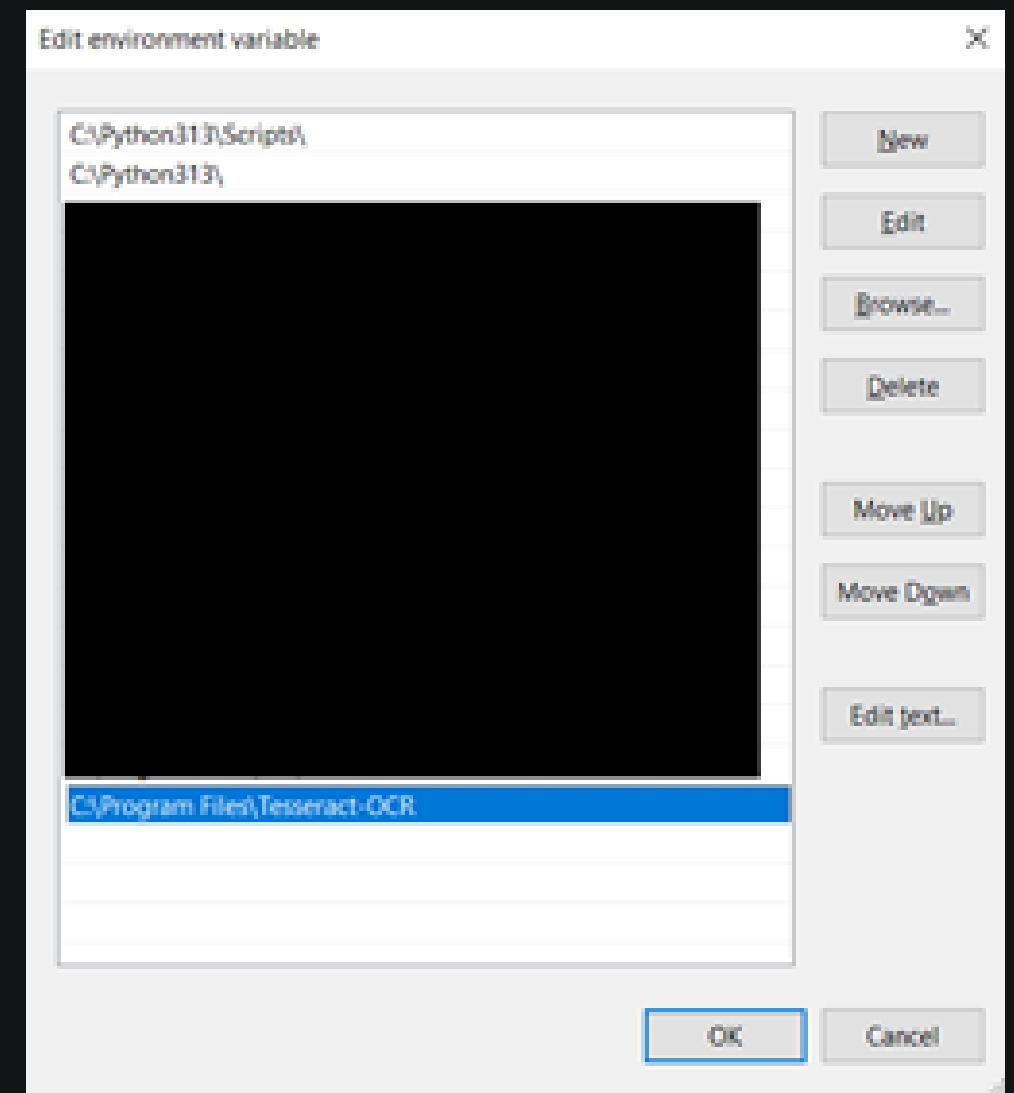
Instalasi dan Persiapan (2) – Perangkat Lunak yang Diperlukan

- Python versi 3.10 atau keatas
 - Unduh dari <https://www.python.org/downloads/>
- Tesseract OCR versi 5.4.0 atau keatas
 - Unduh dari <https://github.com/UB-Mannheim/tesseract/releases/tag/v5.4.0.20240606>
- VSCode sebagai IDE, dan Ekstensi:
 - Python
 - Jupyter
 - Pylance

→ Penjelasan lebih rinci tersedia di manual

Instalasi dan Persiapan (3) – Menginstal Tesseract OCR

- Langkah-langkah:
 - Unduh & jalankan penginstal
 - Ikuti petunjuk persiapan
 - Tambahkan ke Variabel Lingkungan
 - Buka Edit variabel lingkungan sistem
 - Buka Variabel Sistem → Jalur → Baru
 - Tambahkan: C:\Program Files\Tesseract-OCR

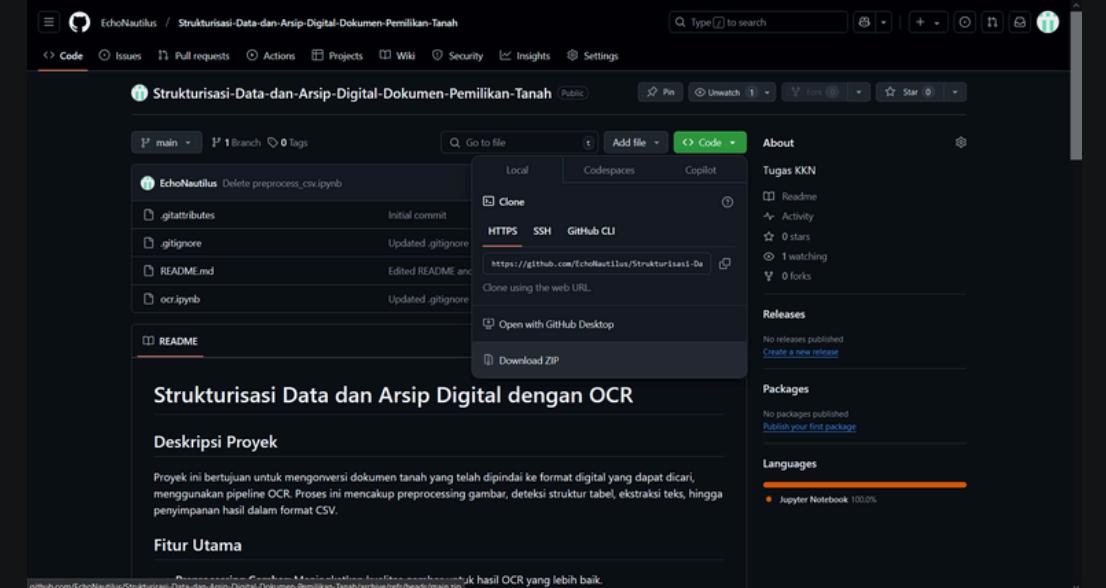


→ Penjelasan lebih rinci tersedia di manual

Kolom *Edit environment variable* yang telah ditambahkan path Tesseract-OCR

Instalasi dan Persiapan (4) – Memperoleh repositori kode

- Langkah-langkah:
 - Unduh zip repo Github dari:
<https://github.com/EchoNautilus/Strukturisasi-Data-dan-Arsip-Digital-Dokumen-Pemilikan-Tanah>
 - Untuk unduh zip file, klik tombol *Code* hijau
 - Klik download zip
 - Ekstraksi zip file dan masukkan file-file PDF kedalamnya.



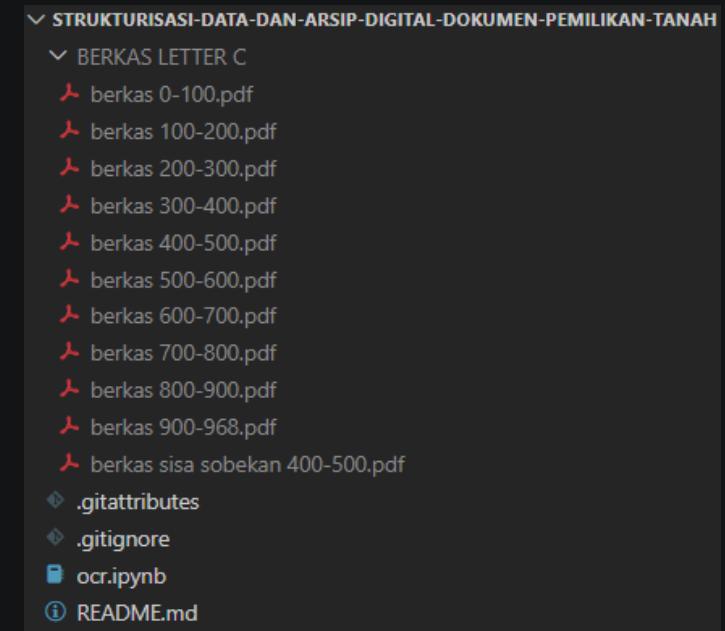
→ Penjelasan lebih rinci tersedia di manual

Tombol unduh zip file

Instalasi dan Persiapan (5) - Folder & Setup File

- Struktur Folder:

```
├── BERKAS LETTER C
│   ├── berkas 0-100.pdf
│   ├── berkas 100-200.pdf
│   ├── ...
│   └── berkas sisa sobekan 400-500.pdf
└── ocr.ipynb
```



Struktur Folder yang Benar

- Catatan penting:

- Pastikan nama file sama persis dengan yang tercantum (peka huruf besar/kecil)
- Jangan mengganti nama atau memindahkan file ke luar struktur
- Periksa ruang penyimpanan (pipeline memerlukan ~6GB, direkomendasi ada 10GB)

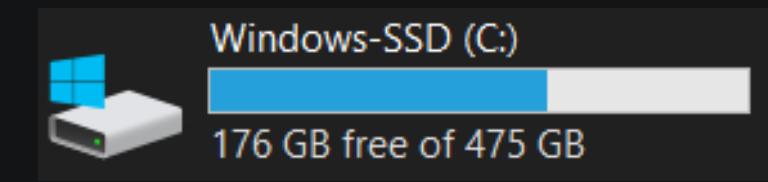
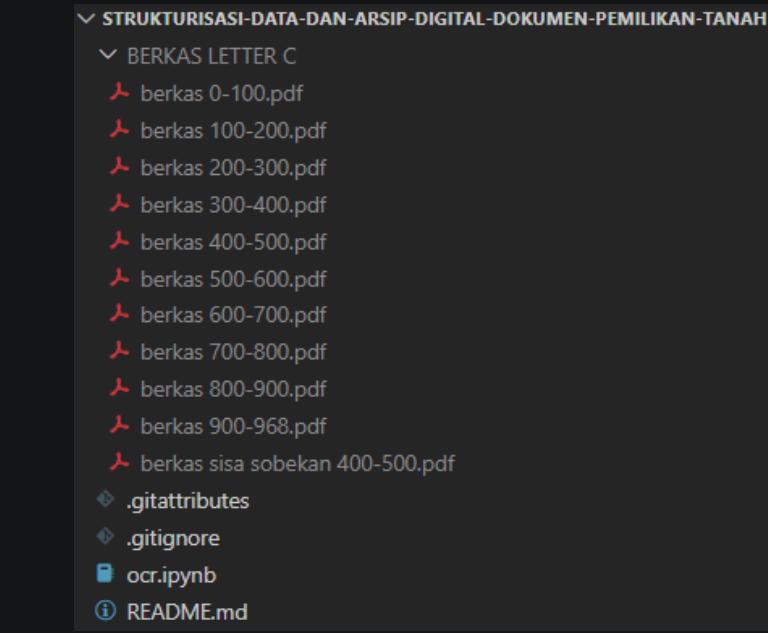
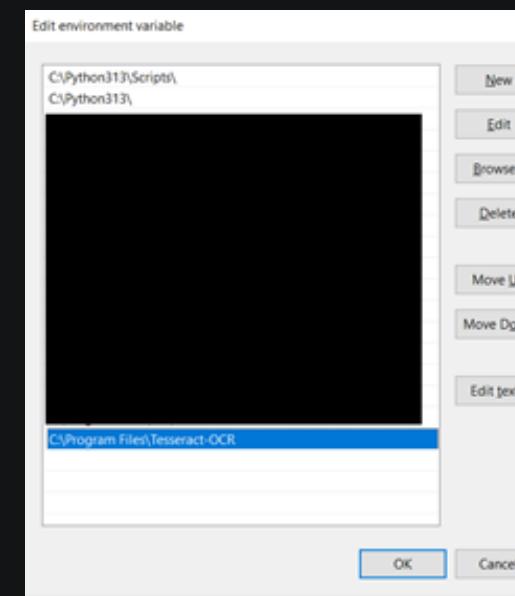
Panduan Penggunaan Langkah Demi Langkah (1) – Daftar Periksa Pra-Jalankan

- Pastikan semua instalasi telah selesai (Python, Tesseract, VSCode)
- Pastikan variabel lingkungan telah ditetapkan (Tesseract ditambahkan ke PATH)
- Verifikasi struktur folder (BERISI HURUF C dengan file yang benar)
- Periksa ruang penyimpanan (minimal 6GB kosong)

```
# Menginstall library-library
%pip install pandas
%pip install pytesseract
%pip install pdf2image
%pip install pillow
%pip install numpy
%pip install matplotlib
%pip install tensorflow
%pip install opencv-python

✓ 16.1s

Requirement already satisfied: pandas in c:\users\...
Requirement already satisfied: numpy<2,>=1.26.0 in ...
Requirement already satisfied: python-dateutil<2...
Requirement already satisfied: pytz>=2020.1 in ...
Requirement already satisfied: tzdata>=2022.7 in ...
Requirement already satisfied: six>=1.5 in c:\use...
Note: you may need to restart the kernel to use t
```



Instalasi package berhasil

Path berada di *Environment Variable*

Struktur Folder yang Benar

Disk memiliki ruang kosong

Panduan Penggunaan Langkah Demi Langkah (2) – Menjalankan Notebook

- 1 Buka ocr.ipynb di VSCode
- 2 Jalankan seluruh notebook (tombol Run All)
- 3 Tunggu pemrosesan (sekitar 30 menit)

Interface pada saat code berjalan

Tombol *Run All* berada di interface atas

The screenshot shows a Jupyter Notebook interface within VSCode. The title bar indicates the file is 'ocr.ipynb'. The main area displays a Python script for processing PDF files. The code includes comments for step 1 (converting PDF to image) and step 2 (processing each page). A progress bar at the bottom shows '8m 40.2s' completed. The status bar at the bottom of the interface shows the 'Run All' button.

```
# Langkah 1: Mengonversi PDF ke Gambar dan Mengatur Folder Penyimpanan
pdf_files = [
    ('BERKAS LETTER C/berkas 0-100.pdf', 'file png berkas 0-100'),
    ('BERKAS LETTER C/berkas 100-200.pdf', 'file png berkas 100-200'),
    ('BERKAS LETTER C/berkas 200-300.pdf', 'file png berkas 200-300'),
    ('BERKAS LETTER C/berkas 300-400.pdf', 'file png berkas 300-400'),
    ('BERKAS LETTER C/berkas 400-500.pdf', 'file png berkas 400-500'),
    ('BERKAS LETTER C/berkas 500-600.pdf', 'file png berkas 500-600'),
    ('BERKAS LETTER C/berkas 600-700.pdf', 'file png berkas 600-700'),
    ('BERKAS LETTER C/berkas 700-800.pdf', 'file png berkas 700-800'),
    ('BERKAS LETTER C/berkas 800-900.pdf', 'file png berkas 800-900'),
    ('BERKAS LETTER C/berkas 900-968.pdf', 'file png berkas 900-968'),
    ('BERKAS LETTER C/berkas sisa sobekan 400-500.pdf', 'file png berkas sisa sobekan 400-500')
]

# Proses setiap file PDF
for pdf_path, output_folder in pdf_files:
    # Pastikan folder output ada
    if not os.path.exists(output_folder):
        os.makedirs(output_folder)

    print(f"Mengonversi {pdf_path} ke gambar...")

    # Konversi PDF ke gambar
    images = convert_from_path(pdf_path)
    for i, image in enumerate(images):
        image_path = os.path.join(output_folder, f'halaman_{i+1}.png') # Tentukan nama file
        image.save(image_path, 'PNG') # Simpan gambar dalam format PNG
        print(f"Gambar halaman {i+1} dari {pdf_path} disimpan di: {image_path}") # Informasi gambar yang disimpan

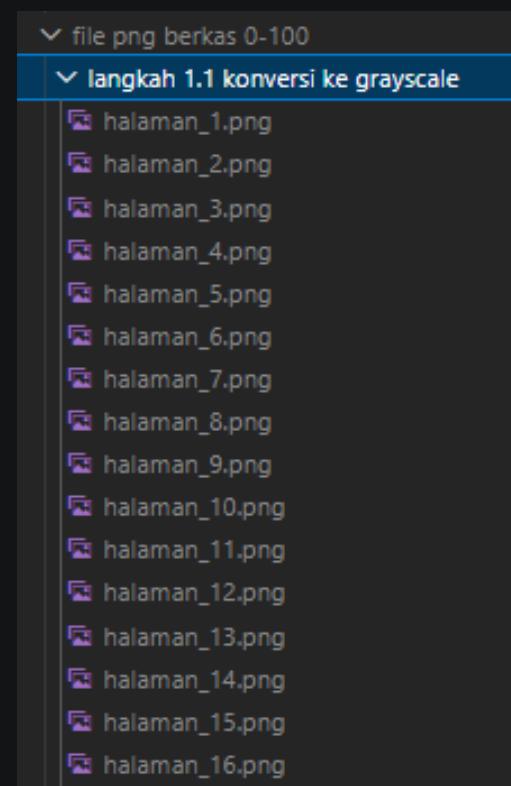
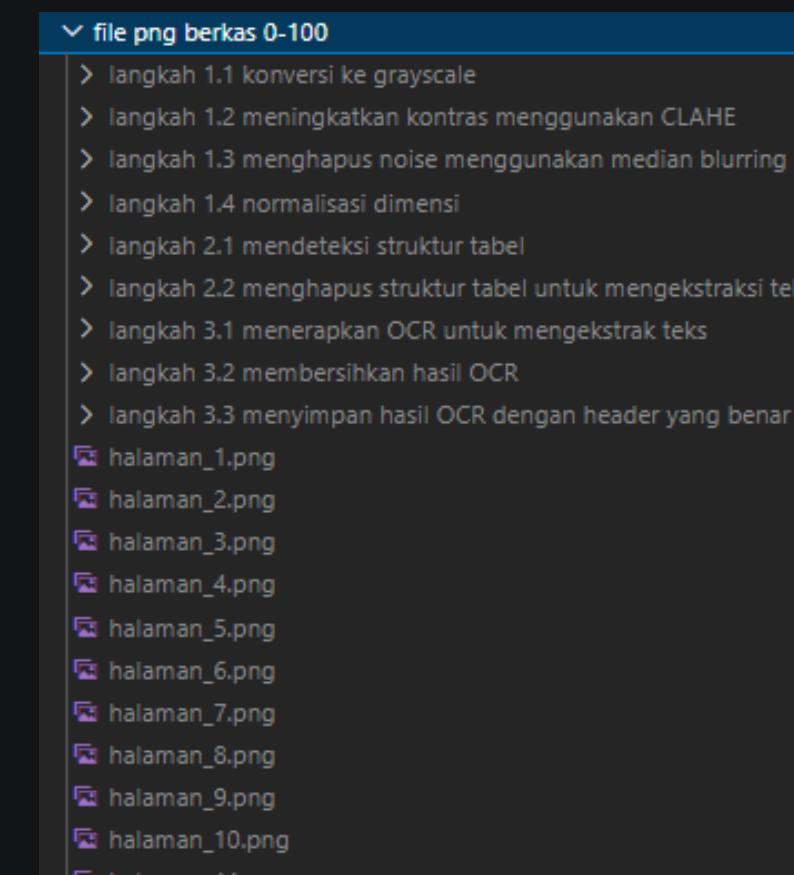
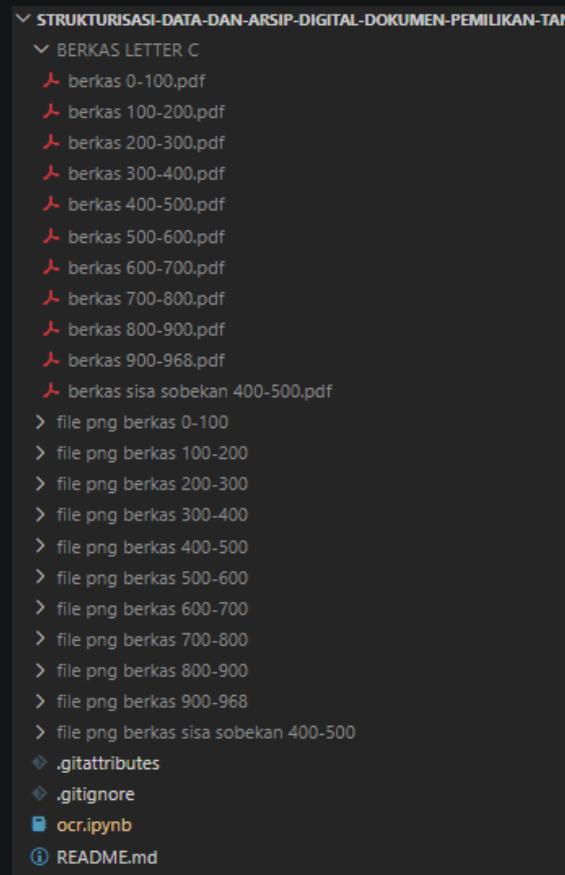
    print("Konversi {pdf_path} selesai!")
print("Semua file PDF selesai diproses!")

... Mengonversi BERKAS LETTER C/berkas 0-100.pdf ke gambar...
Gambar halaman 1 dari BERKAS LETTER C/berkas 0-100.pdf disimpan di: file png berkas 0-100\halaman_1.png
Gambar halaman 2 dari BERKAS LETTER C/berkas 0-100.pdf disimpan di: file png berkas 0-100\halaman_2.png
Gambar halaman 3 dari BERKAS LETTER C/berkas 0-100.pdf disimpan di: file png berkas 0-100\halaman_3.png
Gambar halaman 4 dari BERKAS LETTER C/berkas 0-100.pdf disimpan di: file png berkas 0-100\halaman_4.png
Gambar halaman 5 dari BERKAS LETTER C/berkas 0-100.pdf disimpan di: file png berkas 0-100\halaman_5.png
Gambar halaman 6 dari BERKAS LETTER C/berkas 0-100.pdf disimpan di: file png berkas 0-100\halaman_6.png
Gambar halaman 7 dari BERKAS LETTER C/berkas 0-100.pdf disimpan di: file png berkas 0-100\halaman_7.png
Gambar halaman 8 dari BERKAS LETTER C/berkas 0-100.pdf disimpan di: file png berkas 0-100\halaman_8.png
Gambar halaman 9 dari BERKAS LETTER C/berkas 0-100.pdf disimpan di: file png berkas 0-100\halaman_9.png
Gambar halaman 10 dari BERKAS LETTER C/berkas 0-100.pdf disimpan di: file png berkas 0-100\halaman_10.png
Gambar halaman 11 dari BERKAS LETTER C/berkas 0-100.pdf disimpan di: file png berkas 0-100\halaman_11.png
```

Kode sedang berjalan

Hasil yang Diharapkan – Output Folder & File

- Setelah menjalankan alur kerja, folder root harus berisi:
 - Gambar yang Diproses (skala abu-abu, kontras ditingkatkan, noise dihilangkan)
 - File Teks yang Diekstrak (output OCR mentah, teks dibersihkan)
 - File CSV Final (data dokumen terstruktur dan dapat dicari)



Output yang diharapkan adalah subdirektori yang diisi dengan gambar PDF ke PNG, langkah-langkah praproses gambar, aplikasi OCR ke file .txt, dan file CSV yang dihasilkan.

FAQ dan Troubleshooting (1) – Masalah Umum

🔍 Apakah saya perlu menginstal setiap dependensi secara manual?

✓ Tidak, menjalankan dua sel pertama di ocr.ipynb akan menginstal semuanya kecuali Tesseract OCR.

🔍 Berapa lama waktu pemrosesan?

✓ ~30 menit pada PC dengan Intel i7, RAM 16GB, SSD. Mesin yang lebih lambat mungkin memerlukan waktu lebih lama.

🔍 Mengapa keluaran OCR tidak akurat?

✓ Kemungkinan penyebabnya:

- Teks memudar, tulisan tangan, atau kualitas pindai buruk
- Parameter praproses yang kurang tepat

Karena hasil OCR yang kurang memuaskan, diharapkan kode ini digunakan sebagai basis untuk penyesuaian parameter OCR dan praproses gambar lebih lanjut

FAQ dan Troubleshooting (2) – Pemecahan Error

⚠️ Tesseract tidak terdeteksi

- ◆ Periksa apakah Tesseract telah diinstal dan ditambahkan ke PATH.

⚠️ Masalah: Program macet atau hang

- ◆ Perbaikan: Jalankan sel satu per satu untuk mengidentifikasi langkah yang gagal.

⚠️ Masalah: Output CSV tidak dihasilkan

- ◆ Perbaikan: Pastikan semua langkah berjalan dengan sukses dan folder output tersedia.

Glossarium

Istilah yang Digunakan dalam Presentasi Ini:

- OCR (*Optical Character Recognition*) – Mengubah gambar teks menjadi teks yang dapat dibaca mesin.
- Prapemrosesan – Langkah-langkah penyempurnaan gambar untuk meningkatkan akurasi OCR.
- CLAHE (*Contrast Limited Adaptive Histogram Equalization*) – Metode untuk meningkatkan kontras gambar.
- CSV (*Comma-Separated Values*) – Format teks terstruktur untuk menyimpan data yang diekstrak.
- Pipeline – Urutan langkah-langkah otomatis untuk memproses data.
- Variabel Lingkungan – Pengaturan sistem yang membantu perangkat lunak menemukan dependensi (misalnya, Tesseract PATH).

Referensi

Perangkat Lunak & *Dependency* Resmi

- Python: python.org
- Tesseract OCR: [GitHub Release](#)
- VSCode: code.visualstudio.com

Repositori Proyek

- GitHub Repo: [Strukturisasi Data dan Arsip Digital dengan OCR](#)

Terima Kasih

