FileCrawler.ai — Dokumentasi & Panduan Pembuatan

> **FileCrawler.ai** adalah aplikasi web enterprise berbasis **CodeIgniter 4** yang memungkinkan pengguna mengunggah file PDF, mengekstrak teksnya di sisi klien, mengirimkan teks beserta prompt ke AI (via OpenRouter), lalu menampilkan, menggabungkan (merge), dan mengunduh hasilnya dalam berbagai format (PDF, DOCX, XLSX, JSON).

Daftar Isi

---

1. Gambaran Umum Sistem

FileCrawler.ai dirancang untuk memproses dokumen PDF secara massal dengan bantuan AI. Pengguna memberikan satu atau beberapa file PDF beserta sebuah *prompt* instruksi, lalu AI akan mengekstrak informasi terstruktur dari dokumen tersebut dan menyajikannya dalam format yang bisa langsung diunduh atau digabungkan.

Contoh Use Case:

- Ekstrak data perusahaan (nama, alamat, telepon) dari direktori PDF
- Ringkasan klausul kontrak dari dokumen legal
- Parsing data produk dari katalog PDF

---

2. Arsitektur & Tech Stack

Layer	Teknologi
Backend Framework	CodeIgniter 4 (PHP 8+)
AI Provider	OpenRouter API (model: DeepSeek R1 / configurable)
PDF Generation	TCPDF
Excel Export	PhpSpreadsheet
Word Export	PhpWord
Database	MySQL / MariaDB
Frontend	Vanilla HTML + Tailwind CSS
PDF Parsing	PDF.js (client-side, v3.4.120)
Icon Library	Lucide Icons

Alasan PDF diparsing di sisi klien (bukan server): PDF.js berjalan di browser sehingga file PDF tidak perlu diunggah ke server. Hanya teks hasil ekstraksi yang dikirim, menghemat bandwidth dan meningkatkan privasi.

---

3. Alur Kerja Aplikasi

```
[User] Pilih PDF (max 4 file, max 4 halaman total)
        |
        v
[Browser / PDF.js] Ekstrak teks dari setiap halaman PDF
        |
        v
```

```
[Browser / JS] Gabungkan semua teks + tambahkan <prompt> separator
        |
        v
[POST /file-crawler/process] Kirim JSON { text, prompt, timestamp }
        |
        v
[FileCrawlerController::process()]
  - Validasi input
  - Buat prompt lengkap dengan aturan format JSON
  - Kirim ke OpenRouter API
        |
        v
[OpenRouter / DeepSeek AI]
  Kembalikan JSON terstruktur: { nama_file, data: [...] }
        |
        v
[FileCrawlerController]
  - Parse & validasi JSON response
  - Generate ID unik (file-xxxx-xxxx-xxxx)
  - Simpan ke database (tb_scrape)
  - Return { success, name, id_file, type }
        |
        v
[Browser] Tampilkan kartu hasil di UI
        |
        v
[User] Pilih aksi:
  A) Download langsung → GET /file-crawler/downloadas/{id}/{format}/{nama}
  B) Centang beberapa → POST /file-crawler/merge → Download hasil merge
  C) Flush → Hapus semua hasil dari sesi
```

---

4. Struktur Database

Tabel: `tb_scrape`

```sql
CREATE TABLE `tb_scrape` (
  `id`         INT(11) UNSIGNED NOT NULL AUTO_INCREMENT,
  `id_file`    VARCHAR(100)     NOT NULL UNIQUE,
  `hasil`      LONGTEXT         NOT NULL,
  `nama_file`  VARCHAR(255)     NOT NULL,
  `type`       VARCHAR(20)      NOT NULL DEFAULT 'json',
  `user`       VARCHAR(100)     NOT NULL DEFAULT '',
  PRIMARY KEY (`id`),
  KEY `idx_user` (`user`),
  KEY `idx_id_file` (`id_file`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

Keterangan kolom:

Kolom	Tipe	Keterangan
`id`	INT AUTO_INCREMENT	Primary key internal
`id_file`	VARCHAR(100)	ID unik file, format: `file-xxxx-xxxx-xxxx` atau `merged_xxx`

Kolom	Tipe	Keterangan
`hasil`	LONGTEXT	Data JSON hasil parsing AI (array of objects)
`nama_file`	VARCHAR(255)	Nama deskriptif file dari AI
`type`	VARCHAR(20)	Format file: `json`, `pdf`, `docx`, `xlsx`
`user`	VARCHAR(100)	Device fingerprint pengguna (untuk isolasi data per user)

> **Catatan:** Saat "Flush", field `user` di-set menjadi string kosong, bukan menghapus baris (soft dissociation).

---

5. Panduan Pembuatan — Backend (PHP/CI4)

5.1 Setup Proyek CodeIgniter 4

```
# Install CI4 via Composer
composer create-project codeigniter4/appstarter filecrawler-ai
cd filecrawler-ai

# Install dependencies tambahan
composer require tecnickcom/tcpdf
composer require phpoffice/phpspreadsheet
composer require phpoffice/phpword
```

Konfigurasi `.env`:

```
# Database
database.default.hostname = localhost
database.default.database = filecrawler_db
database.default.username = root
database.default.password = yourpassword
database.default.DBDriver = MySQLi

# OpenRouter
OPENROUTER_API_KEY = sk-or-v1-xxxxxxxxxxxxxxxx
OPENROUTER_MODEL   = deepseek/deepseek-r1-0528:free
```

5.2 Library OpenRouter

Buat file `app/Libraries/OpenRouter.php`. Library ini adalah HTTP client untuk OpenRouter API dengan dukungan:

- **Non-streaming** (`complete()`) — untuk proses biasa
- **Streaming** (`stream()`, `streamComplete()`) — untuk respons real-time
- **Config normalization** — konversi camelCase ke snake_case

Kode Lengkap Beserta Penjelasan Inline Code:

```
<?php

namespace App\Libraries;
```

```php
use Exception;

/**
 * Class OpenRouter
 *
 * Library untuk berkomunikasi dengan OpenRouter API — sebuah gateway
 * yang menyediakan akses ke berbagai model AI (GPT, Claude, DeepSeek, dll)
 * melalui satu endpoint yang seragam.
 */
class OpenRouter
{
    // Menyimpan API Key yang diambil dari environment (.env)
    private string $apiKey;

    // Menyimpan nama model AI yang akan digunakan, contoh: 'openai/gpt-4o'
    private string $model;

    // Batas waktu maksimum (dalam detik) untuk menunggu respons dari API
    // Jika lebih dari 60 detik tidak ada respons, request akan dibatalkan
    private int $timeout = 60;

    // URL endpoint API OpenRouter yang digunakan untuk semua permintaan chat
    private string $baseUrl = "https://openrouter.ai/api/v1/chat/completions";

    // Menyimpan konfigurasi default yang berlaku untuk semua request
    // Contoh: temperature, max_tokens, dll — bisa diset sekali lalu dipakai berulang
    private array $defaultConfig = [];

    /**
     * Konstruktor — dipanggil saat membuat objek baru: new OpenRouter()
     *
     * @param ?string $model  Nama model yang ingin digunakan (opsional)
     *                        Jika tidak diisi, akan dibaca dari .env OPENROUTER_MODEL
     *                        Jika .env juga kosong, pakai model default DeepSeek
     */
    public function __construct(?string $model = null)
    {
        // Ambil API Key dari environment variable, jika tidak ada set string kosong
        $this->apiKey = getenv('OPENROUTER_API_KEY') ?: '';

        // Jika API Key kosong, lempar error — tidak bisa lanjut tanpa API Key
        if (!$this->apiKey) {
            throw new Exception("Missing OPENROUTER_API_KEY in .env");
        }

        // Tentukan model yang dipakai:
        // Prioritas 1: parameter $model yang dioper saat new OpenRouter('nama-model')
        // Prioritas 2: variabel OPENROUTER_MODEL di file .env
        // Prioritas 3 (default): deepseek/deepseek-r1-0528:free
        $this->model = $model ?: getenv('OPENROUTER_MODEL') ?: 'deepseek/deepseek-r1-
0528:free';
    }

    /**
     * Menetapkan konfigurasi default yang akan digunakan di semua request.
     * Berguna agar tidak perlu mengulang parameter yang sama setiap kali memanggil
complete() atau stream().
     *
```

```
     * Contoh penggunaan:
     *   $router->setGenerationConfig(['temperature' => 0.7, 'maxTokens' => 500]);
     *
     * @param array $config  Array konfigurasi dengan key camelCase (contoh: 'maxTokens',
'topP')
     * @return self          Mengembalikan objek ini sendiri supaya bisa di-chain:
     *                       $router->setGenerationConfig([...])->complete($messages)
     */
    public function setGenerationConfig(array $config): self
    {
        // Simpan konfigurasi setelah dikonversi ke format snake_case yang dimengerti API
        $this->defaultConfig = $this->normalizeConfig($config);
        return $this;
    }

    /**
     * Mengambil konfigurasi default yang sedang aktif.
     * Berguna untuk debugging atau mengecek konfigurasi yang sudah diset.
     *
     * @return array  Konfigurasi saat ini dalam format snake_case
     */
    public function getGenerationConfig(): array
    {
        return $this->defaultConfig;
    }

    /**
     * Menghapus semua konfigurasi default, mengembalikannya ke kondisi awal (array
kosong).
     * Gunakan ini jika ingin "reset" konfigurasi setelah sebelumnya diset.
     *
     * @return self  Mengembalikan objek ini sendiri untuk mendukung method chaining
     */
    public function resetGenerationConfig(): self
    {
        $this->defaultConfig = [];
        return $this;
    }

    /**
     * Mengirim pesan ke API dan menunggu SELURUH respons selesai sebelum dikembalikan.
     * Cocok untuk kasus di mana kamu butuh teks lengkapnya dulu sebelum diproses.
     *
     * Cara kerja: kirim request → tunggu → terima teks utuh → return
     *
     * @param array $messages  Array percakapan, contoh:
     *                         [['role' => 'user', 'content' => 'Halo!']]
     * @param array $options   Parameter tambahan khusus untuk request ini.
     *                         Jika sama dengan defaultConfig, nilai $options yang menang.
     * @return string          Isi teks dari respons model AI
     */
    public function complete(array $messages, array $options = []): string
    {
        // Gabungkan konfigurasi default dengan options yang dioper,
        // options akan menimpa (override) jika ada key yang sama
        $mergedOptions = array_merge($this->defaultConfig, $this-
>normalizeConfig($options));

        // Susun payload lengkap (model + messages + semua parameter)
```

```
        $payload = $this->buildPayload($messages, $mergedOptions);

        // Kirim HTTP POST dan simpan hasilnya sebagai array
        $res = $this->post($this->baseUrl, $payload);

        // Ambil teks dari struktur respons API:
        // $res['choices'][0]['message']['content'] → isi pesan dari model
        // Jika struktur tidak ada (error, dll), kembalikan string kosong
        return $res['choices'][0]['message']['content'] ?? '';
    }

    /**
     * Mengirim pesan ke API dan memproses respons secara STREAMING (real-time).
     * Teks dari model dikirim sepotong-sepotong (chunk), dan callback dipanggil
     * setiap kali ada potongan baru — sehingga teks bisa langsung ditampilkan
     * ke user tanpa menunggu semuanya selesai.
     *
     * Cocok untuk: tampilan chat real-time, Server-Sent Events (SSE), CLI output
langsung.
     *
     * @param array    $messages  Array percakapan
     * @param callable $callback  Fungsi yang dipanggil tiap chunk:
     *                            function(string $chunk, bool $isDone)
     *                            - $chunk   : potongan teks yang baru diterima
     *                            - $isDone  : true jika streaming sudah selesai
     * @param array    $options   Parameter tambahan
     */
    public function stream(array $messages, callable $callback, array $options = []): void
    {
        // Gabungkan default config + options, lalu tambahkan 'stream' => true
        // agar API tahu bahwa kita minta respons secara streaming (Server-Sent Events)
        $mergedOptions = array_merge($this->defaultConfig, $this-
>normalizeConfig($options));
        $payload = $this->buildPayload($messages, array_merge($mergedOptions, ['stream' =>
true]));

        // Buat HTTP context untuk membuka koneksi streaming menggunakan fopen()
        // Berbeda dengan post() yang memakai file_get_contents() dan menunggu respons
penuh
        $ctx = stream_context_create([
            "http" => [
                "method"  => "POST",
                "header"  =>
                    // Header autentikasi dengan API Key
                    "Authorization: Bearer {$this->apiKey}\r\n" .
                    // Beritahu server bahwa kita kirim JSON
                    "Content-Type: application/json\r\n",
                // Encode payload ke JSON untuk dikirim sebagai body request
                "content"       => json_encode($payload),
                "timeout"       => $this->timeout,
                // ignore_errors: true → agar fopen() tidak gagal saat status HTTP error
(4xx/5xx)
                // sehingga kita tetap bisa membaca pesan errornya
                "ignore_errors" => true
            ]
        ]);

        // Buka koneksi streaming ke API — fopen() di sini tidak menunggu seluruh respons,
        // melainkan membuka "saluran" yang bisa dibaca baris per baris secara langsung
```

```
        $stream = fopen($this->baseUrl, 'r', false, $ctx);

        // Jika gagal membuka koneksi (misalnya masalah jaringan), lempar error
        if ($stream === false) {
            throw new Exception("Failed to open stream");
        }

        try {
            // Baca stream baris per baris selama koneksi belum habis (feof = end of file)
            while (!feof($stream)) {
                // Baca satu baris dari stream
                $line = fgets($stream);

                // Jika gagal baca (misal timeout sesaat), lewati dan coba lagi
                if ($line === false) {
                    continue;
                }

                // Hapus whitespace/newline di awal dan akhir baris
                $line = trim($line);

                // Baris kosong adalah pemisah antar event di format SSE — lewati saja
                if (empty($line)) {
                    continue;
                }

                // Format SSE (Server-Sent Events): setiap data diawali dengan "data: "
                // Kita hanya proses baris yang dimulai dengan prefix ini
                if (strpos($line, 'data: ') === 0) {
                    // Ambil bagian setelah "data: " (mulai dari karakter ke-6)
                    $data = substr($line, 6);

                    // "[DONE]" adalah sinyal dari API bahwa streaming telah selesai
                    if ($data === '[DONE]') {
                        // Panggil callback dengan chunk kosong dan isDone = true sebagai
tanda selesai
                        $callback('', true);
                        break; // Keluar dari loop
                    }

                    // Parse JSON dari data yang diterima
                    $json = json_decode($data, true);

                    // Cek apakah ada konten teks di dalam struktur delta (potongan baru)
                    // Struktur: choices[0].delta.content berisi teks yang baru saja
digenerate
                    if (isset($json['choices'][0]['delta']['content'])) {
                        $chunk = $json['choices'][0]['delta']['content'];
                        // Kirim potongan teks ini ke callback dengan isDone = false
                        $callback($chunk, false);
                    }

                    // Cek juga apakah model sudah selesai generate (finish_reason tidak
null)
                    // Ini adalah cara alternatif API memberi tahu bahwa streaming selesai
                    if (
                        isset($json['choices'][0]['finish_reason']) &&
                        $json['choices'][0]['finish_reason'] !== null
                    ) {
```

```
                        $callback('', true);
                        break; // Keluar dari loop
                    }
                }
            }
        } finally {
            // Blok finally selalu dijalankan, baik ada error maupun tidak
            // Pastikan koneksi stream selalu ditutup untuk membebaskan resource
            fclose($stream);
        }
    }

    /**
     * Menggabungkan streaming dengan pengumpulan teks secara otomatis.
     * Streaming tetap berjalan di balik layar (sehingga callback bisa dipakai
     * untuk real-time display), tapi fungsi ini juga mengumpulkan semua chunk
     * dan mengembalikan teks lengkap di akhir.
     *
     * Keuntungan: dapat dua hal sekaligus — tampilan real-time + teks utuh untuk
diproses.
     *
     * @param array         $messages  Array percakapan
     * @param callable|null $onChunk   Callback opsional per chunk (boleh null jika tidak
butuh real-time)
     * @param array         $options   Parameter tambahan
     * @return string                  Teks lengkap hasil gabungan semua chunk
     */
    public function streamComplete(array $messages, ?callable $onChunk = null, array
$options = []): string
    {
        // Variabel penampung teks yang dikumpulkan dari semua chunk
        // Menggunakan reference (&$fullText) agar bisa dimodifikasi dari dalam closure
        $fullText = '';

        // Panggil stream() dengan callback internal yang mengumpulkan teks
        $this->stream($messages, function ($chunk, $isDone) use (&$fullText, $onChunk) {
            if (!$isDone) {
                // Tambahkan chunk ke teks yang sedang dikumpulkan
                $fullText .= $chunk;

                // Jika ada callback eksternal, teruskan chunk ke sana juga
                if ($onChunk) {
                    $onChunk($chunk, $isDone);
                }
            } elseif ($onChunk) {
                // Streaming selesai — teruskan sinyal selesai ke callback eksternal jika
ada
                $onChunk('', $isDone);
            }
        }, $options);

        // Kembalikan keseluruhan teks yang sudah terkumpul
        return $fullText;
    }

    /**
     * Mengkonversi key array dari format camelCase ke snake_case sesuai standar API.
     *
     * Mengapa perlu? API OpenRouter menggunakan snake_case (contoh: max_tokens, top_p),
```

```
     * tapi lebih nyaman menulis camelCase di PHP (contoh: maxTokens, topP).
     * Fungsi ini menjembatani keduanya agar developer bisa pakai gaya PHP yang natural.
     *
     * Key yang tidak ada di mapping akan dibiarkan apa adanya (mungkin sudah snake_case).
     *
     * @param array $config  Config dengan key camelCase
     * @return array         Config dengan key snake_case siap dikirim ke API
     */
    private function normalizeConfig(array $config): array
    {
        $normalized = [];

        // Tabel pemetaan: camelCase (PHP) → snake_case (API)
        $mapping = [
            'temperature'        => 'temperature',        // Kreativitas output (0.0 -
2.0)
            'topP'               => 'top_p',              // Nucleus sampling
            'topK'               => 'top_k',              // Batasi kandidat token teratas
            'frequencyPenalty'   => 'frequency_penalty',  // Kurangi pengulangan kata yang
sering
            'presencePenalty'    => 'presence_penalty',   // Dorong topik baru
            'repetitionPenalty'  => 'repetition_penalty', // Penalti pengulangan kalimat
            'minP'               => 'min_p',              // Ambang minimum probabilitas
token
            'topA'               => 'top_a',              // Alternatif nucleus sampling
            'seed'               => 'seed',               // Untuk output yang
deterministik (reproducible)
            'maxTokens'          => 'max_tokens',         // Batas panjang output
            'logitBias'          => 'logit_bias',         // Bias manual per token
            'logprobs'           => 'logprobs',           // Aktifkan log probabilitas
            'topLogprobs'        => 'top_logprobs',       // Jumlah logprob teratas yang
dikembalikan
            'responseFormat'     => 'response_format',    // Format respons (misal: JSON
mode)
            'structuredOutputs'  => 'structured_outputs', // Structured output schema
            'stop'               => 'stop',               // Kata/kalimat yang
menghentikan generasi
            'tools'              => 'tools',              // Definisi fungsi untuk tool
calling
            'toolChoice'         => 'tool_choice',        // Kontrol kapan model pakai
tool
            'parallelToolCalls'  => 'parallel_tool_calls',// Izinkan pemanggilan tool
secara paralel
            'verbosity'          => 'verbosity',          // Tingkat detail output
            'model'              => 'model',              // Override model untuk request
ini
            'stream'             => 'stream'              // Flag streaming (diset
otomatis oleh stream())
        ];

        foreach ($config as $key => $value) {
            // Jika key ada di mapping, gunakan versi snake_case-nya
            if (isset($mapping[$key])) {
                $normalized[$mapping[$key]] = $value;
            }
            // Jika tidak ada di mapping, biarkan key-nya apa adanya
            // (kemungkinan sudah snake_case atau parameter kustom)
            else {
                $normalized[$key] = $value;
```

```
            }
        }

        return $normalized;
    }

    /**
     * Menyusun payload lengkap yang akan dikirim ke API sebagai body request JSON.
     * Hanya parameter yang benar-benar diset yang dimasukkan ke payload
     * (tidak mengirim nilai null atau default yang tidak perlu).
     *
     * Setiap parameter juga divalidasi tipe dan rentang nilainya agar
     * tidak mengirim data yang akan ditolak API.
     *
     * @param array $messages  Array pesan percakapan
     * @param array $options   Parameter konfigurasi yang sudah dinormalisasi (snake_case)
     * @return array           Payload siap kirim
     */
    private function buildPayload(array $messages, array $options = []): array
    {
        // Dua field wajib: model yang digunakan dan array pesan percakapan
        // Jika $options mengandung 'model', gunakan itu sebagai override
        $payload = [
            'model'    => $options['model'] ?? $this->model,
            'messages' => $messages
        ];

        // --- PARAMETER SAMPLING ---
        // Setiap parameter ditambahkan ke payload hanya jika diset,
        // dengan validasi tipe data dan rentang nilai yang diizinkan

        $this->addIfSet($payload, $options, 'temperature', 'float', 0.0, 2.0);       //
Kreativitas
        $this->addIfSet($payload, $options, 'top_p', 'float', 0.0, 1.0);             //
Nucleus sampling
        $this->addIfSet($payload, $options, 'top_k', 'int', 0);                      //
Top-K sampling
        $this->addIfSet($payload, $options, 'frequency_penalty', 'float', -2.0, 2.0);//
Penalti frekuensi
        $this->addIfSet($payload, $options, 'presence_penalty', 'float', -2.0, 2.0); //
Penalti kehadiran
        $this->addIfSet($payload, $options, 'repetition_penalty', 'float', 0.0, 2.0);//
Penalti repetisi
        $this->addIfSet($payload, $options, 'min_p', 'float', 0.0, 1.0);             //
Min probability
        $this->addIfSet($payload, $options, 'top_a', 'float', 0.0, 1.0);             //
Top-A sampling

        // --- PARAMETER GENERASI ---
        $this->addIfSet($payload, $options, 'seed', 'int');           // Seed untuk
reproducibility
        $this->addIfSet($payload, $options, 'max_tokens', 'int', 1);  // Minimal 1 token
output

        // --- LOGIT BIAS ---
        // logit_bias harus berupa array (key: token ID, value: bias -100 sampai 100)
        // jadi kita validasi manual bahwa nilainya memang array
        if (isset($options['logit_bias']) && is_array($options['logit_bias'])) {
            $payload['logit_bias'] = $options['logit_bias'];
```

```
        }

        $this->addIfSet($payload, $options, 'logprobs', 'bool');              // Log
probabilitas
        $this->addIfSet($payload, $options, 'top_logprobs', 'int', 0, 20);   // Max 20
logprobs

        // --- FORMAT RESPONS ---
        // response_format bisa berupa array kompleks (misal: {'type': 'json_object'})
        // sehingga langsung dimasukkan tanpa konversi tipe tambahan
        if (isset($options['response_format'])) {
            $payload['response_format'] = $options['response_format'];
        }

        $this->addIfSet($payload, $options, 'structured_outputs', 'bool'); // Structured
outputs

        // --- STOP SEQUENCES ---
        // API menerima string tunggal atau array string sebagai stop sequence
        // Kita normalisasi: jika string, bungkus jadi array satu elemen
        if (isset($options['stop'])) {
            $payload['stop'] = is_array($options['stop']) ? $options['stop'] :
[$options['stop']];
        }

        // --- TOOL CALLING ---
        // tools harus array (list definisi fungsi), validasi tipe sebelum memasukkan
        if (isset($options['tools']) && is_array($options['tools'])) {
            $payload['tools'] = $options['tools'];
        }

        // tool_choice bisa string ('auto', 'none') atau array, langsung masukkan apa
adanya
        if (isset($options['tool_choice'])) {
            $payload['tool_choice'] = $options['tool_choice'];
        }

        $this->addIfSet($payload, $options, 'parallel_tool_calls', 'bool'); // Parallel
tool calls

        // --- VERBOSITY ---
        // Hanya nilai yang valid ('low', 'medium', 'high', 'max') yang diterima
        // Nilai lain diabaikan untuk mencegah error API
        if (
            isset($options['verbosity']) &&
            in_array($options['verbosity'], ['low', 'medium', 'high', 'max'])
        ) {
            $payload['verbosity'] = $options['verbosity'];
        }

        // --- STREAM FLAG ---
        // Jika diset, konversi ke boolean eksplisit (true/false)
        // Flag ini diset otomatis oleh method stream() dan streamComplete()
        if (isset($options['stream'])) {
            $payload['stream'] = (bool)$options['stream'];
        }

        return $payload;
    }
```

```php
    /**
     * Helper untuk menambahkan satu parameter ke payload HANYA jika parameter tersebut
ada di $options.
     * Selain itu, fungsi ini juga mengkonversi tipe data dan memastikan nilai berada
dalam rentang yang valid.
     *
     * Mengapa dipisah jadi fungsi sendiri? Karena buildPayload() punya banyak parameter
     * dengan logika yang sama (cek ada → konversi tipe → clamp rentang → tambah ke
payload).
     * Memisahkannya menghindari pengulangan kode (DRY principle).
     *
     * @param array  &$payload  Payload yang sedang disusun (passed by reference agar bisa
dimodifikasi)
     * @param array   $options  Array opsi yang diperiksa
     * @param string  $key      Nama parameter (snake_case)
     * @param string  $type     Tipe data: 'float', 'int', atau 'bool'
     * @param mixed   $min      Nilai minimum yang diizinkan (null = tidak ada batas
bawah)
     * @param mixed   $max      Nilai maksimum yang diizinkan (null = tidak ada batas
atas)
     */
    private function addIfSet(array &$payload, array $options, string $key, string $type,
$min = null, $max = null): void
    {
        // Jika parameter tidak ada di options, tidak perlu lakukan apa-apa
        if (!isset($options[$key])) {
            return;
        }

        $value = $options[$key];

        // Konversi tipe data dan terapkan batas rentang (clamping)
        switch ($type) {
            case 'float':
                $value = (float)$value;
                // Pastikan nilai tidak di bawah minimum
                if ($min !== null && $value < $min) $value = $min;
                // Pastikan nilai tidak di atas maksimum
                if ($max !== null && $value > $max) $value = $max;
                break;

            case 'int':
                $value = (int)$value;
                if ($min !== null && $value < $min) $value = $min;
                if ($max !== null && $value > $max) $value = $max;
                break;

            case 'bool':
                // Konversi ke boolean (0, '', null, 'false' semua jadi false; sisanya
true)
                $value = (bool)$value;
                break;
        }

        // Masukkan nilai yang sudah divalidasi ke dalam payload
        $payload[$key] = $value;
    }
```

```
    /**
     * Mengirim HTTP POST request ke API dan mengembalikan respons sebagai array PHP.
     * Digunakan oleh method complete() untuk request non-streaming.
     *
     * Menggunakan file_get_contents() yang menunggu SELURUH respons sebelum lanjut,
     * berbeda dengan fopen() di stream() yang membaca baris per baris secara real-time.
     *
     * @param string $url      URL endpoint yang dituju
     * @param array  $payload  Data yang akan dikirim sebagai JSON body
     * @return array           Respons API yang sudah di-decode dari JSON ke array PHP
     */
    private function post(string $url, array $payload): array
    {
        // Buat HTTP context dengan konfigurasi request
        $ctx = stream_context_create([
            "http" => [
                "method"  => "POST",
                "header"  =>
                    // Header autentikasi — API Key dikirim sebagai Bearer token
                    "Authorization: Bearer {$this->apiKey}\r\n" .
                    // Beritahu server bahwa body request berformat JSON
                    "Content-Type: application/json\r\n",
                // Encode payload PHP array ke string JSON untuk dikirim
                "content"       => json_encode($payload),
                "timeout"       => $this->timeout,
                // ignore_errors: true → jangan throw error saat HTTP 4xx/5xx
                // agar kita masih bisa membaca body respons error dari API
                "ignore_errors" => true
            ]
        ]);

        // Kirim request dan tunggu seluruh respons (blocking call)
        $res = file_get_contents($url, false, $ctx);

        // Jika file_get_contents() gagal total (bukan error HTTP, tapi koneksi gagal)
        if ($res === false) {
            throw new Exception("HTTP request failed");
        }

        // Parse string JSON menjadi array PHP
        $decoded = json_decode($res, true);

        // Jika API mengembalikan field 'error', berarti ada masalah di sisi API
        // Lempar Exception dengan pesan error dari API
        if (isset($decoded['error'])) {
            throw new Exception("API Error: " . ($decoded['error']['message'] ?? 'Unknown
error'));
        }

        // Kembalikan array hasil decode, atau array kosong jika decode gagal
        return $decoded ?? [];
    }
}
```

Konfigurasi yang direkomendasikan untuk ekstraksi data terstruktur:

```php
$this->openrouter->setGenerationConfig([
    'temperature' => 0.0,   // Deterministik, tidak ada variasi
    'topP'        => 0.1,   // Fokus pada token dengan probabilitas tertinggi
    'topK'        => 1,     // Hanya token terbaik
    'seed'        => 42     // Reproducible output
]);
```

5.2.1 Referensi Parameter Lengkap

Library mendukung 22+ parameter yang dapat diteruskan melalui `setGenerationConfig()` maupun parameter `$options` di setiap pemanggilan method. Key menggunakan format **camelCase** dan secara internal akan dikonversi ke snake_case sebelum dikirim ke API.

camelCase (input)	snake_case (API)	Tipe	Range / Nilai Valid	Keterangan
`temperature`	`temperature`	float	0.0 – 2.0	Kreativitas output. 0.0 = deterministik
`topP`	`top_p`	float	0.0 – 1.0	Nucleus sampling. Gunakan bersamaan dengan temperature
`topK`	`top_k`	int	0+	Batasi pilihan token teratas
`frequencyPenalty`	`frequency_penalty`	float	-2.0 – 2.0	Penalti token yang sering muncul
`presencePenalty`	`presence_penalty`	float	-2.0 – 2.0	Penalti token yang sudah muncul
`repetitionPenalty`	`repetition_penalty`	float	0.0 – 2.0	Penalti pengulangan kalimat
`minP`	`min_p`	float	0.0 – 1.0	Minimum probability threshold
`topA`	`top_a`	float	0.0 – 1.0	Top-A sampling
`seed`	`seed`	int	bebas	Seed untuk output reproducible
`maxTokens`	`max_tokens`	int	min 1	Batas maksimal token output
`logitBias`	`logit_bias`	array	`[token_id => bias]`	Bias terhadap token tertentu
`logprobs`	`logprobs`	bool	true/false	Aktifkan log-probabilities di response
`topLogprobs`	`top_logprobs`	int	0 – 20	Jumlah top logprobs yang dikembalikan
`responseFormat`	`response_format`	array	`['type' => 'json_object']`	Paksa format response tertentu
`structuredOutputs`	`structured_outputs`	bool	true/false	Aktifkan structured outputs

camelCase (input)	snake_case (API)	Tipe	Range / Nilai Valid	Keterangan
`stop`	`stop`	string/array	string atau array of string	Stop sequence — hentikan generasi saat string ini muncul
`tools`	`tools`	array	OpenAI tool schema	Definisi tool/function calling
`toolChoice`	`tool_choice`	string/array	`'auto'`, `'none'`, atau object	Kontrol pemilihan tool
`parallelToolCalls`	`parallel_tool_calls`	bool	true/false	Izinkan pemanggilan tool secara paralel
`verbosity`	`verbosity`	string	`'low'`, `'medium'`, `'high'`, `'max'`	Tingkat verbositas response
`model`	`model`	string	nama model OpenRouter	Override model untuk request ini saja
`stream`	`stream`	bool	true/false	Aktifkan streaming (dikelola otomatis oleh method `stream()`)

> **Catatan:** Parameter yang tidak disetel tidak akan dikirimkan ke API. Library hanya mengirim parameter yang secara eksplisit di-set, sehingga nilai default API berlaku untuk parameter yang tidak disebutkan.

5.2.2 Manajemen Konfigurasi

Library menyediakan tiga method untuk mengelola konfigurasi generasi:

```php
// Set konfigurasi default yang berlaku untuk semua request berikutnya
$openrouter->setGenerationConfig([
    'temperature' => 0.0,
    'topP'        => 0.1,
    'maxTokens'   => 2000,
]);

// Baca konfigurasi yang sedang aktif (berguna untuk debugging)
$currentConfig = $openrouter->getGenerationConfig();
// Mengembalikan array dengan key snake_case, contoh:
// ['temperature' => 0.0, 'top_p' => 0.1, 'max_tokens' => 2000]

// Reset semua konfigurasi ke default (kosong)
// Berguna saat perlu mengirim request bersih tanpa config sebelumnya
$openrouter->resetGenerationConfig();
```

Ketiga method ini bersifat **chainable**:

```php
$openrouter
    ->resetGenerationConfig()
```

```
    ->setGenerationConfig(['temperature' => 0.7])
    ->complete($messages);
```

5.2.3 Streaming

Library menyediakan dua method streaming untuk respons real-time via **Server-Sent Events (SSE)**:

`stream()` **— Streaming dengan callback per chunk:**

Cocok untuk mengirim output langsung ke browser secara bertahap tanpa menunggu respons selesai.

```php
// Contoh: stream output ke browser secara real-time
header('Content-Type: text/event-stream');
header('Cache-Control: no-cache');

$messages = [['role' => 'user', 'content' => $prompt]];

$this->openrouter->stream(
    $messages,
    function(string $chunk, bool $isDone) {
        if (!$isDone) {
            echo "data: " . json_encode(['chunk' => $chunk]) . "\n\n";
            ob_flush();
            flush();
        } else {
            echo "data: [DONE]\n\n";
            ob_flush();
            flush();
        }
    },
    ['temperature' => 0.7]  // options opsional, override defaultConfig
);
```

`streamComplete()` **— Streaming + kumpulkan full text:**

Cocok saat butuh teks lengkap di akhir sekaligus ingin memantau progress per chunk.

```php
$messages = [['role' => 'user', 'content' => $prompt]];

// Tanpa callback — hanya ambil teks lengkap
$fullText = $this->openrouter->streamComplete($messages);

// Dengan callback — pantau progress sekaligus ambil hasil akhir
$fullText = $this->openrouter->streamComplete(
    $messages,
    function(string $chunk, bool $isDone) {
        if (!$isDone) {
            // Opsional: forward ke SSE client atau log progress
            log_message('debug', 'Chunk received: ' . $chunk);
        }
    },
    ['temperature' => 0.7]  // options opsional
);
```

```
// $fullText berisi seluruh teks yang terkumpul
echo $fullText;
```

Perbedaan `stream()` vs `streamComplete()`:

	`stream()`	`streamComplete()`
Return value	`void`	`string` (teks lengkap)
Callback	Wajib	Opsional
Use case	Real-time SSE ke browser	Butuh hasil akhir + monitoring

---

5.2.4 Override Model Per-Request

Model dapat diganti untuk request tertentu saja tanpa mengubah konfigurasi global atau membuat instance baru, dengan meneruskan `model` di parameter `$options`:

```
// Instance default menggunakan model dari .env (misal: deepseek/deepseek-r1-0528:free)
$openrouter = new OpenRouter();

// Override model hanya untuk request ini
$response = $openrouter->complete($messages, [
    'model' => 'openai/gpt-4o'
]);

// Request berikutnya kembali ke model default
$response2 = $openrouter->complete($messages);

// Override juga berlaku untuk stream
$openrouter->stream($messages, $callback, [
    'model' => 'anthropic/claude-3-haiku'
]);
```

Model yang tersedia dapat dilihat di openrouter.ai/models.

5.2.5 Error Handling

Library melempar `Exception` dengan tiga kondisi yang berbeda. Semua pemanggilan ke library sebaiknya dibungkus `try-catch`:

```
try {
    $response = $this->openrouter->complete($messages);
} catch (\Exception $e) {
    $message = $e->getMessage();

    if (str_starts_with($message, 'API Error:')) {
        // Error dari OpenRouter API — misal: model tidak tersedia,
        // API key tidak valid, rate limit, dsb.
        // Pesan asli dari API sudah disertakan di $message
        log_message('error', 'OpenRouter API error: ' . $message);
```

```
        return $this->response->setJSON(['success' => false, 'error' => $message])-
>setStatusCode(502);

    } elseif ($message === 'HTTP request failed') {
        // Koneksi ke OpenRouter gagal — misal: timeout, DNS error, jaringan down
        log_message('error', 'Network error reaching OpenRouter');
        return $this->response->setJSON(['success' => false, 'error' => 'Koneksi ke AI
gagal'])->setStatusCode(503);

    } elseif ($message === 'Failed to open stream') {
        // Khusus method stream() — gagal membuka koneksi SSE
        log_message('error', 'Failed to open SSE stream to OpenRouter');
        return $this->response->setJSON(['success' => false, 'error' => 'Streaming tidak
tersedia'])->setStatusCode(503);

    } else {
        // Error lainnya (misal: Missing API key saat konstruksi)
        log_message('error', 'Unexpected OpenRouter error: ' . $message);
        return $this->response->setJSON(['success' => false, 'error' => 'Internal
error'])->setStatusCode(500);
    }
}
```

5.3 FileCrawlerController

Buat file `app/Controllers/FileCrawlerController.php`.

Method `process()` — Inti Sistem

Ini adalah endpoint paling krusial. Berikut penjelasan setiap langkah:

Langkah 1 — Terima Input:

```
$json = $this->request->getJSON(true);
$prompt = $json['prompt'];
$fullTextFile = $json['text'];
```

Langkah 2 — Buat Full Prompt dengan Aturan Ketat:

Prompt dirancang dengan aturan eksplisit agar AI selalu mengembalikan JSON valid:

```
$fullPrompt = <<<PROMPT
$fullTextFile

[ATURAN]
1. Respon WAJIB tanpa markdown format
2. Respon tanpa pembuka dan penutup, HANYA HASIL YANG DIMINTA
3. JSON harus valid dan bisa diparse oleh json_decode()
4. Beri value "nama_file" dari kesimpulan isi data
5. Tidak boleh ada object tanpa key

FORMAT:
{"nama_file": "value", "data": [{"field1": "value", ...}]}
```

```
[INSTRUKSI]
$prompt
PROMPT;
```

Langkah 3 — Kirim ke AI dan Parse Response:

```
$response = $this->openrouter->complete($aiPrompt);

// Fix edge case: jika response berakhir dengan ']' tanpa '}'
if (substr($response, -1) === ']') {
    $response .= '}';
}

// Bersihkan BOM dan encoding issues
$response = trim($response);
$response = preg_replace('/^\xEF\xBB\xBF/', '', $response);
$response = mb_convert_encoding($response, 'UTF-8', 'UTF-8');

$json_decode = json_decode($response, true);

if (json_last_error() !== JSON_ERROR_NONE) {
    throw new Exception(json_last_error_msg());
}
```

Langkah 4 — Generate ID dan Simpan ke DB:

```
$id_file = sprintf(
    'file-%04x%04x-%04x',
    random_int(0, 0xffff),
    random_int(0, 0x0fff) | 0x4000,
    random_int(0, 0x3fff) | 0x8000,
);

$user = $_COOKIE['device_fp'] ?? "";

$modelScrape = new ScrapeModel();
$modelScrape->insert([
    'id_file'    => $id_file,
    'hasil'      => json_encode($json_decode['data']),
    'nama_file'  => $json_decode['nama_file'],
    'type'       => 'json',
    'user'       => $user
]);
```

Method `getFiles()` — Load File Saat Halaman Dibuka

```
public function getFiles()
{
    $user = $_COOKIE['device_fp'] ?? "";
    if (empty($user)) {
        return $this->response->setJSON(['success' => false, 'error' => 'Unauthorized'])-
```

```
>setStatusCode(401);
    }

    $modelScrape = new ScrapeModel();
    $files = $modelScrape->where("user", $user)->select('id_file, type, nama_file')-
>findAll();

    return $this->response->setJSON(['success' => true, 'files' => $files, 'total' =>
count($files)]);
}
```

Method `flush()` — Hapus Sesi User

Daripada menghapus data secara permanen, `flush()` hanya men-set field `user` menjadi string kosong:

```
foreach ($files as $file) {
    $modelScrape->update($file['id'], ['user' => '']);
}
```

5.4 DownloadController

Buat file `app/Controllers/DownloadController.php`. Controller ini menangani konversi data JSON dari database ke berbagai format file.

Alur Utama `downloadAs()`

```
public function downloadAs($name, $ext, $fileNames)
{
    // 1. Ambil data JSON dari database
    $jsonData = $this->getJSONData($name);

    // 2. Decode JSON
    $data = json_decode($jsonData, true);

    // 3. Sanitize filename
    $fileName = $this->sanitizeFileName($fileNames);

    // 4. Dispatch ke method yang sesuai
    switch (strtolower($ext)) {
        case 'pdf':  $this->downloadAsPDF($data, $fileName);   break;
        case 'xlsx': $this->downloadAsExcel($data, $fileName); break;
        case 'docx': $this->downloadAsWord($data, $fileName);  break;
        case 'json': $this->downloadAsJSON($data, $fileName);  break;
    }
}
```

Helper `getHeaders()` — Ekstrak Kolom Dinamis

Terdapat dua implementasi `getHeaders` di controller ini dengan perilaku yang berbeda:

Versi aktif — `getHeaders()` (gabungkan semua baris):

```php
private function getHeaders(array $data): array
{
    $allKeys = [];
    foreach ($data as $item) {
        $allKeys = array_merge($allKeys, array_keys((array) $item));
    }
    $uniqueKeys = array_unique($allKeys);

    $headers = [];
    foreach ($uniqueKeys as $key) {
        $headers[$key] = $this->keyToLabel($key);
    }
    return $headers;
}
```

Versi alternatif — `getHeaders_1()` (hanya baris pertama):

```php
private function getHeaders_1(array $data): array
{
    $keys = array_keys($data[0]);
    $headers = [];
    foreach ($keys as $key) {
        $headers[$key] = $this->keyToLabel($key);
    }
    return $headers;
}
```

Kapan menggunakan masing-masing:

	`getHeaders()`	`getHeaders_1()`
Sumber key	Semua baris digabung	Hanya baris pertama
Aman untuk data tidak konsisten	☑ Ya	❌ Bisa kehilangan kolom
Performa	Sedikit lebih lambat	Lebih cepat
Rekomendasi	**Gunakan ini untuk export**	Hanya untuk data yang sudah pasti konsisten

> Menggunakan **semua key dari semua baris** memastikan tidak ada kolom yang terlewat jika baris berbeda memiliki field yang berbeda.

Helper `keyToLabel()` — Konversi Key ke Label

Digunakan oleh `getHeaders()` untuk mengubah format `snake_case` menjadi `Title Case` yang readable:

```php
private function keyToLabel(string $key): string
{
    $label = str_replace('_', ' ', $key);  // "nama_perusahaan" → "nama perusahaan"
    $label = ucwords($label);               // → "Nama Perusahaan"
    return $label;
}
```

Helper `sanitizeFileName()` — Sanitasi Nama File

Membersihkan nama file sebelum digunakan sebagai nama output download, melalui empat langkah:

```
private function sanitizeFileName(string $fileName): string
{
    $fileName = preg_replace('/\.[^.]+$/', '', $fileName);      // Hapus ekstensi (.pdf, .docx, dll)
    $fileName = preg_replace('/[^a-zA-Z0-9_-]/', '_', $fileName); // Karakter non-alphanumeric → underscore
    $fileName = preg_replace('/_+/', '_', $fileName);           // Collapse multiple underscore → satu
    $fileName = trim($fileName, '_');                           // Hapus underscore di awal/akhir

    return $fileName ?: 'download';  // Fallback jika hasil kosong
}
```

Contoh transformasi: `"Data Perusahaan Jakarta.pdf"` → `"Data_Perusahaan_Jakarta"`

Export PDF (TCPDF)

```
private function downloadAsPDF(array $data, string $fileName)
{
    $headers = $this->getHeaders($data);

    $pdf = new TCPDF(PDF_PAGE_ORIENTATION, PDF_UNIT, PDF_PAGE_FORMAT, true, 'UTF-8', false);
    $pdf->setPrintHeader(false);
    $pdf->setPrintFooter(false);
    $pdf->SetMargins(15, 15, 15);
    $pdf->SetAutoPageBreak(TRUE, 15);
    $pdf->AddPage();

    // Judul dokumen di tengah halaman
    $pdf->SetFont('helvetica', 'B', 18);
    $pdf->Cell(0, 12, strtoupper($fileName), 0, 1, 'C');
    $pdf->Ln(3);

    // Garis horizontal dekoratif
    $pdf->SetDrawColor(50, 50, 50);
    $pdf->Line(15, $pdf->GetY(), 195, $pdf->GetY());
    $pdf->Ln(8);

    foreach ($data as $index => $item) {
        // Auto page break saat mendekati batas halaman (Y > 250)
        if ($pdf->GetY() > 250) {
            $pdf->AddPage();
        }

        // Field pertama dipakai sebagai judul item, tidak diulang di detail
        $firstKey = array_key_first($item);
        $pdf->SetFont('helvetica', 'B', 13);
        $pdf->Cell(0, 8, ($index + 1) . '. ' . $item[$firstKey], 0, 1, 'L');
```

```
        $skipFirst = false;
        foreach ($item as $key => $value) {
            if (!$skipFirst) { $skipFirst = true; continue; } // Skip field pertama

            // Field dengan nilai kosong atau '-' tidak ditampilkan
            if (!empty($value) && $value !== '-') {
                $pdf->SetX(20); // Indentasi 20pt
                $pdf->SetFont('helvetica', 'B', 10);
                $pdf->Cell(45, 6, $headers[$key], 0, 0, 'L');
                $pdf->SetFont('helvetica', '', 10);
                $pdf->Cell(5, 6, ':', 0, 0, 'C');
                $pdf->MultiCell(0, 6, $value, 0, 'L');
            }
        }

        $pdf->Ln(6);
        // Garis pemisah ringan antar item
        $pdf->SetDrawColor(200, 200, 200);
        $pdf->Line(20, $pdf->GetY(), 190, $pdf->GetY());
        $pdf->Ln(6);
    }

    // PENTING: Bersihkan buffer sebelum output
    while (ob_get_level()) { ob_end_clean(); }

    $pdf->Output($fileName . '.pdf', 'D');
    exit;
}
```

Export Excel (PhpSpreadsheet)

```
private function downloadAsExcel(array $data, string $fileName)
{
    $headers = $this->getHeaders($data);
    $keys    = array_keys($headers);

    $spreadsheet = new Spreadsheet();
    $sheet = $spreadsheet->getActiveSheet();
    $sheet->setTitle('Data Export');

    // Tulis header row
    $col = 1;
    foreach ($headers as $key => $label) {
        $cellAddress =
\PhpOffice\PhpSpreadsheet\Cell\Coordinate::stringFromColumnIndex($col) . '1';
        $sheet->setCellValue($cellAddress, $label);
        $col++;
    }

    $lastCol =
\PhpOffice\PhpSpreadsheet\Cell\Coordinate::stringFromColumnIndex(count($headers));

    // Styling header: biru (#4472C4) dengan teks putih, bold, center, border
    $sheet->getStyle('A1:' . $lastCol . '1')->applyFromArray([
        'font'      => ['bold' => true, 'color' => ['rgb' => 'FFFFFF'], 'size' => 11],
        'fill'      => ['fillType' => Fill::FILL_SOLID, 'startColor' => ['rgb' =>
```

```
'4472C4']],
        'alignment' => ['horizontal' => Alignment::HORIZONTAL_CENTER, 'vertical' =>
Alignment::VERTICAL_CENTER],
        'borders'   => ['allBorders' => ['borderStyle' => Border::BORDER_THIN, 'color' =>
['rgb' => '000000']]]
    ]);

    // Tinggi header row 25px
    $sheet->getRowDimension(1)->setRowHeight(25);

    // Tulis data rows
    $row = 2;
    foreach ($data as $item) {
        $col = 1;
        foreach ($keys as $key) {
            $cellAddress =
\PhpOffice\PhpSpreadsheet\Cell\Coordinate::stringFromColumnIndex($col) . $row;
            $sheet->setCellValue($cellAddress, $item[$key] ?? '');
            $col++;
        }

        // Border tipis pada setiap sel data
        $sheet->getStyle('A' . $row . ':' . $lastCol . $row)->applyFromArray([
            'borders'   => ['allBorders' => ['borderStyle' => Border::BORDER_THIN, 'color'
=> ['rgb' => 'CCCCCC']]],
            'alignment' => ['vertical' => Alignment::VERTICAL_TOP]
        ]);

        // Alternating row color — baris genap abu-abu (#F2F2F2)
        if ($row % 2 == 0) {
            $sheet->getStyle('A' . $row . ':' . $lastCol . $row)
                  ->getFill()->setFillType(Fill::FILL_SOLID)
                  ->getStartColor()->setRGB('F2F2F2');
        }
        $row++;
    }

    // Auto-size semua kolom agar konten tidak terpotong
    for ($i = 1; $i <= count($headers); $i++) {
        $colLetter = \PhpOffice\PhpSpreadsheet\Cell\Coordinate::stringFromColumnIndex($i);
        $sheet->getColumnDimension($colLetter)->setAutoSize(true);
    }

    // Freeze baris header agar tetap terlihat saat scroll
    $sheet->freezePane('A2');

    while (ob_get_level()) { ob_end_clean(); }
    header('Content-Type: application/vnd.openxmlformats-
officedocument.spreadsheetml.sheet');
    header('Content-Disposition: attachment; filename="' . $fileName . '.xlsx"');
    header('Cache-Control: max-age=0');
    header('Pragma: public');
    (new Xlsx($spreadsheet))->save('php://output');
    exit;
}
```

Export Word (PhpWord)

```
private function downloadAsWord(array $data, string $fileName)
{
    $headers = $this->getHeaders($data);
    $phpWord = new PhpWord();

    // Section dengan margin kustom (1000 twips ≈ 1.76cm semua sisi)
    $section = $phpWord->addSection([
        'marginTop' => 1000, 'marginBottom' => 1000,
        'marginLeft' => 1000, 'marginRight' => 1000
    ]);

    // Judul dokumen di tengah, warna biru (#1F497D)
    $section->addText(strtoupper($fileName), [
        'name' => 'Arial', 'size' => 18, 'bold' => true, 'color' => '1F497D'
    ], ['alignment' => Jc::CENTER, 'spaceAfter' => 300]);

    // Garis horizontal dekoratif
    $section->addLine(['weight' => 1, 'width' => 450, 'height' => 0, 'color' =>
'1F497D']);
    $section->addTextBreak(1);

    // Tiga style terpisah untuk konsistensi tampilan
    $titleStyle = ['name' => 'Arial', 'size' => 13, 'bold' => true, 'color' => '1F497D'];
    $labelStyle = ['name' => 'Arial', 'size' => 10, 'bold' => true];
    $valueStyle = ['name' => 'Arial', 'size' => 10];

    foreach ($data as $index => $item) {
        // Field pertama sebagai judul item
        $firstKey = array_key_first($item);
        $section->addText(($index + 1) . '. ' . ($item[$firstKey] ?? ''), $titleStyle,
['spaceAfter' => 200]);

        $skipFirst = false;
        foreach ($item as $key => $value) {
            if (!$skipFirst) { $skipFirst = true; continue; }
            if (!empty($value) && $value !== '-') {
                $textRun = $section->addTextRun(['spaceAfter' => 100]);
                $textRun->addText($headers[$key] . ': ', $labelStyle);
                $textRun->addText($value, $valueStyle);
            }
        }

        $section->addTextBreak(1);

        // Garis pemisah antar item, KECUALI item terakhir
        if ($index < count($data) - 1) {
            $section->addLine(['weight' => 0.5, 'width' => 450, 'height' => 0, 'color' =>
'CCCCCC']);
            $section->addTextBreak(1);
        }
    }

    $objWriter = IOFactory::createWriter($phpWord, 'Word2007');
    while (ob_get_level()) { ob_end_clean(); }
    header('Content-Type: application/vnd.openxmlformats-
officedocument.wordprocessingml.document');
    header('Content-Disposition: attachment; filename="' . $fileName . '.docx"');
```

```
    header('Cache-Control: max-age=0');
    header('Pragma: public');
    $objWriter->save('php://output');
    exit;
}
```

Export JSON

```
private function downloadAsJSON(array $data, string $fileName)
{
    // JSON_PRETTY_PRINT      : Format indentasi agar mudah dibaca
    // JSON_UNESCAPED_UNICODE  : Karakter non-ASCII (é, ñ, dsb.) tidak di-escape
    // JSON_UNESCAPED_SLASHES  : Slash (/) tidak di-escape menjadi \/
    $jsonContent = json_encode($data, JSON_PRETTY_PRINT | JSON_UNESCAPED_UNICODE |
JSON_UNESCAPED_SLASHES);

    while (ob_get_level()) { ob_end_clean(); }
    header('Content-Type: application/json; charset=UTF-8');
    header('Content-Disposition: attachment; filename="' . $fileName . '.json"');
    header('Cache-Control: no-cache, must-revalidate');
    header('Pragma: public');
    echo $jsonContent;
    exit;
}
```

> **Catatan `JSON_UNESCAPED_UNICODE`:** Flag ini penting untuk data berbahasa Indonesia agar karakter seperti huruf berdiakritik tidak berubah menjadi escape sequence `\uXXXX` di dalam file JSON hasil download.

Method `merge()` — Gabungkan Beberapa File

```
public function merge()
{
    $json    = $this->request->getJSON();
    $fileIds = $json->file_ids;
    $format  = $json->format;

    // Gabungkan semua data dari file-file yang dipilih
    $data_list = [];
    foreach ($fileIds as $fi) {
        $jsonData = $this->getJSONData($fi);
        $data = json_decode($jsonData, true);
        if (is_array($data)) {
            array_push($data_list, ...$data); // Spread operator (PHP 7.4+)
        }
    }

    // Buat file merge baru
    $mergedId = uniqid('merged_', true);
    $modelScrape = new ScrapeModel();
    $modelScrape->insert([
        'id_file'   => $mergedId,
        'hasil'     => json_encode($data_list),
        'nama_file' => $mergedId,
```

```
            'type'      => $format,
            'user'      => $_COOKIE['device_fp'] ?? ""
        ]);

        // ⚠ PENTING: File-file sumber DIHAPUS PERMANEN dari database setelah merge
        $modelScrape->whereIn('id_file', $fileIds)->delete();

        return $this->response->setJSON([
            'success'   => true,
            'merged_id' => $mergedId,
            'format'    => $format,
            'file_merge' => $fileIds
        ]);
    }
```

> ⚠ **Peringatan:** Berbeda dengan `flush()` yang hanya melakukan *soft dissociation* (mengosongkan field `user`), method `merge()` **menghapus baris secara permanen** dari tabel `tb_scrape` untuk semua `id_file` sumber yang diikutsertakan dalam merge. Operasi ini tidak dapat dibatalkan. Pastikan UI memberi tahu pengguna sebelum melakukan merge bahwa file-file asal akan hilang dan digantikan oleh satu file hasil gabungan.
>
> Selain itu, `nama_file` untuk hasil merge diisi dengan nilai `uniqid('merged_', true)` yang berupa string unik berbasis timestamp — bukan nama yang readable. Frontend perlu menampilkan atau mengganti nama ini sesuai kebutuhan UX.

5.5 ScrapeModel

```
// app/Models/ScrapeModel.php
namespace App\Models;

use CodeIgniter\Model;

class ScrapeModel extends Model
{
    protected $table         = 'tb_scrape';
    protected $primaryKey    = 'id';
    protected $returnType    = 'array';
    protected $allowedFields = ['id_file', 'hasil', 'user', 'type', 'nama_file'];
    protected $useTimestamps = false;
}
```

5.6 Konfigurasi Routes

```
// app/Config/Routes.php
$routes->group('file-crawler', function ($routes) {
    $routes->get('/',                                   'FileCrawlerController::index');
    $routes->post('file-upload',
'FileCrawlerController::fileUpload');
    $routes->post('process',
'FileCrawlerController::process');
    $routes->post('merge',                              'DownloadController::merge');
    $routes->get('get-files',
```

```
'FileCrawlerController::getFiles');
    $routes->get('flush',                                   'FileCrawlerController::flush');
    $routes->delete('delete-file/(:segment)',
'FileCrawlerController::deleteFile/$1');
    $routes->get('downloadas/(:segment)/(:segment)/(:segment)',
'DownloadController::downloadAs/$1/$2/$3');
});
```

---

6. Panduan Pembuatan — Frontend (HTML/JS)

6.1 Ekstraksi PDF dengan PDF.js

PDF.js dijalankan seluruhnya di browser, sehingga tidak ada upload file ke server.

```
<!-- Import via CDN -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/pdf.js/3.4.120/pdf.min.js"></script>

<script>
// Set worker
pdfjsLib.GlobalWorkerOptions.workerSrc =
    'https://cdnjs.cloudflare.com/ajax/libs/pdf.js/3.4.120/pdf.worker.min.js';

// Fungsi ekstraksi teks dari satu file PDF
async function extractPDFText(file) {
    const arrayBuffer = await file.arrayBuffer(); // Baca file sebagai binary
    const pdf = await pdfjsLib.getDocument(arrayBuffer).promise;
    let fullText = "";

    for (let i = 1; i <= pdf.numPages; i++) {
        const page = await pdf.getPage(i);
        const content = await page.getTextContent();
        // Gabungkan semua string di halaman tersebut
        const strings = content.items.map(item => item.str);
        fullText += strings.join(' ');
    }
    return fullText;
}
</script>
```

6.2 Validasi File

Validasi dilakukan di dua tahap:

Tahap 1 — Validasi Jumlah File:

```
fileInput.onchange = async (e) => {
    const files = Array.from(e.target.files);

    if (files.length > 4) {
        showError('Upload Gagal', 'Maksimal 4 file sekaligus.', 'warning');
        e.target.value = '';
        return;
```

```
    }
    // ...
};
```

Tahap 2 — Validasi Halaman Per File:

```
async function checkPageCount(file) {
    const arrayBuffer = await file.arrayBuffer();
    const pdf = await pdfjsLib.getDocument(arrayBuffer).promise;
    const totalPages = pdf.numPages;

    // Cek batas per file
    if (totalPages > 4) {
        showError('File Terlalu Panjang',
            `Maksimal 4 halaman.<br/>- ${file.name} (${totalPages} halaman)`, 'warning');
        return false;
    }

    // Cek batas total kumulatif
    const currentTotal = totalPageFile.reduce((acc, val) => acc + val, 0);
    if (currentTotal + totalPages > 4) {
        showError('Upload Dibatalkan',
            `Total halaman akan melebihi 4. Saat ini sudah ${currentTotal} halaman.`,
'info');
        return false;
    }

    totalPageFile.push(totalPages);
    return true;
}
```

State yang dikelola:

```
let arrayFiles = [];      // File objects yang valid
let totalPageFile = [];   // Jumlah halaman per file (untuk kalkulasi total)
let fullTextParse = "";   // Teks gabungan dari semua PDF
```

6.3 Pengiriman ke Backend

```
document.getElementById('processBtn').onclick = async () => {
    try {
        overlay.classList.remove('hidden'); // Tampilkan loading

        // Ekstrak teks dari semua file secara berurutan
        for (const af of arrayFiles) {
            const extractedText = await extractPDFText(af);
            fullTextParse += extractedText;
        }

        // Kirim ke backend (gunakan separator <prompt> untuk pisahkan teks & prompt)
        const aiResult = await sendToBackend(
```

```
            fullTextParse + "<prompt>" + document.getElementById('promptInput').value
        );

        allResults.push(...aiResult);
        updateUI();
    } catch (e) {
        showError('Processing Error', e.message, 'error');
    } finally {
        overlay.classList.add('hidden');
    }
};

async function sendToBackend(rawText) {
    const API_ENDPOINT = "/file-crawler/process";
    const spliter = rawText.split("<prompt>");

    const response = await fetch(API_ENDPOINT, {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({
            text: spliter[0],       // Teks PDF
            timestamp: new Date().toISOString(),
            prompt: spliter[1]      // Instruksi pengguna
        })
    });

    const data = await response.json();
    if (data.success) {
        return [data]; // Return sebagai array untuk push ke allResults
    } else {
        throw new Error(data.error || 'Proses gagal dilakukan');
    }
}
```

> **Mengapa menggunakan separator `<prompt>`?** Ini cara sederhana untuk memisahkan konten PDF dengan instruksi pengguna dalam satu string sebelum di-split di frontend, menghindari kompleksitas tambahan.

6.4 Manajemen State & UI

```
// State global
let selectedResults = new Set(); // ID kartu yang dipilih untuk merge
let allResults = [];             // Semua kartu hasil (dari DB atau proses baru)

// Render semua kartu hasil
function updateUI() {
    const container = document.getElementById('resultContainer');
    container.innerHTML = allResults.map(res => `
        <div id="card-${res.id_file}" class="result-card ...">
            <!-- Checkbox untuk seleksi merge -->
            <input type="checkbox"
                   id="checkbox-${res.id_file}"
                   class="result-checkbox"
                   onchange="toggleResultSelection('${res.id_file}')">

            <!-- Info file -->
```

```
            <h4>${res.name}</h4>
            <p>${res.type}</p>

            <!-- Dropdown download -->
            <div class="dropdown">
                <button>Download ▼</button>
                <div class="dropdown-content">
                    <button onclick="downloadAs('${res.id_file}', 'pdf',
'${res.name}')">Export PDF</button>
                    <button onclick="downloadAs('${res.id_file}', 'xlsx',
'${res.name}')">Export XLSX</button>
                    <!-- dst. -->
                </div>
            </div>
        </div>
    `).join('');
}

// Toggle seleksi kartu
function toggleResultSelection(id) {
    const checkbox = document.getElementById(`checkbox-${id}`);
    const card = document.getElementById(`card-${id}`);

    if (selectedResults.has(id)) {
        selectedResults.delete(id);
        checkbox.checked = false;
        card.classList.remove('selected'); // Hapus highlight biru
    } else {
        selectedResults.add(id);
        checkbox.checked = true;
        card.classList.add('selected');    // Tambah highlight biru
    }

    updateMergeButton(); // Tampilkan/sembunyikan tombol merge
}
```

6.5 Fitur Merge & Download

Merge:

```
async function mergeSelectedAs(format) {
    if (selectedResults.size === 0) return;

    const response = await fetch('/file-crawler/merge', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({
            file_ids: Array.from(selectedResults),
            format: format
        })
    });

    const data = await response.json();

    if (data.success) {
```

```
        // Hapus kartu yang sudah di-merge dari UI
        const idSet = new Set(data.file_merge);
        allResults = allResults.filter(r => !idSet.has(r.id_file));

        // Tambahkan kartu merge baru
        allResults.push({
            name: data.merged_id,
            id_file: data.merged_id,
            type: data.format
        });

        selectedResults.clear();
        updateUI();
    }
}
```

Download:

```
function downloadAs(name, format, fileName) {
    // Navigasi langsung ke URL download — browser akan handle download
    window.location.href = `/file-crawler/downloadas/${name}/${format}/${fileName}`;
}
```

6.6 Device Fingerprinting

Digunakan untuk mengidentifikasi pengguna tanpa login, sehingga setiap pengguna hanya melihat file miliknya sendiri.

```
function generateFingerprint() {
    // Gabungkan berbagai properti browser yang unik
    const canvas = document.createElement("canvas");
    const ctx = canvas.getContext("2d");
    ctx.textBaseline = "top";
    ctx.font = "14px Arial";
    ctx.fillText("fingerprint", 2, 2);

    const components = [
        navigator.userAgent,
        navigator.language,
        screen.colorDepth,
        screen.width + "x" + screen.height,
        new Date().getTimezoneOffset(),
        !!window.sessionStorage,
        !!window.localStorage,
        canvas.toDataURL()           // Canvas fingerprint (sangat unik per device)
    ];

    return hashString(components.join("|||"));
}

// Simpan ke cookie (1 tahun)
function saveFingerprintToCookie() {
    const fingerprint = generateFingerprint();
    const expires = new Date();
```

```
    expires.setFullYear(expires.getFullYear() + 1);
    document.cookie = `device_fp=${fingerprint}; expires=${expires.toUTCString()}; path=/;
SameSite=Strict`;
    return fingerprint;
}

// Baca fingerprint (atau buat baru jika belum ada)
function getFingerprint() {
    const match = document.cookie.match(/device_fp=([^;]+)/);
    return match ? match[1] : saveFingerprintToCookie();
}
```

> **Penting:** Fingerprinting ini bukan mekanisme autentikasi yang aman untuk data sensitif. Ini hanya untuk memisahkan sesi antar pengguna dalam konteks yang non-kritis.

---

7. Endpoint API

Method	Endpoint	Controller::Method	Keterangan
GET	/file-crawler/	FileCrawlerController::index	Halaman utama
POST	/file-crawler/process	FileCrawlerController::process	Proses teks + prompt → AI
GET	/file-crawler/get-files	FileCrawlerController::getFiles	Ambil daftar file user
GET	/file-crawler/flush	FileCrawlerController::flush	Hapus semua file sesi
POST	/file-crawler/merge	DownloadController::merge	Merge beberapa file
GET	/file-crawler/downloadas/{id}/{format}/{nama}	DownloadController::downloadAs	Download file
POST	/file-crawler/file-upload	FileCrawlerController::fileUpload	Upload file (opsional)
DELETE	/file-crawler/delete-file/{id}	FileCrawlerController::deleteFile	Hapus satu file

Request/Response Format

POST `/file-crawler/process`

```
// Request
{
  "text": "Teks yang diekstrak dari PDF...",
  "prompt": "Ekstrak nama perusahaan, alamat, dan nomor telepon",
```

```
  "timestamp": "2024-01-15T10:30:00.000Z"
}

// Response sukses
{
  "success": true,
  "name": "Data Perusahaan Jakarta",
  "id_file": "file-4a2b3c4d-5e6f-7a8b",
  "type": "json"
}
```

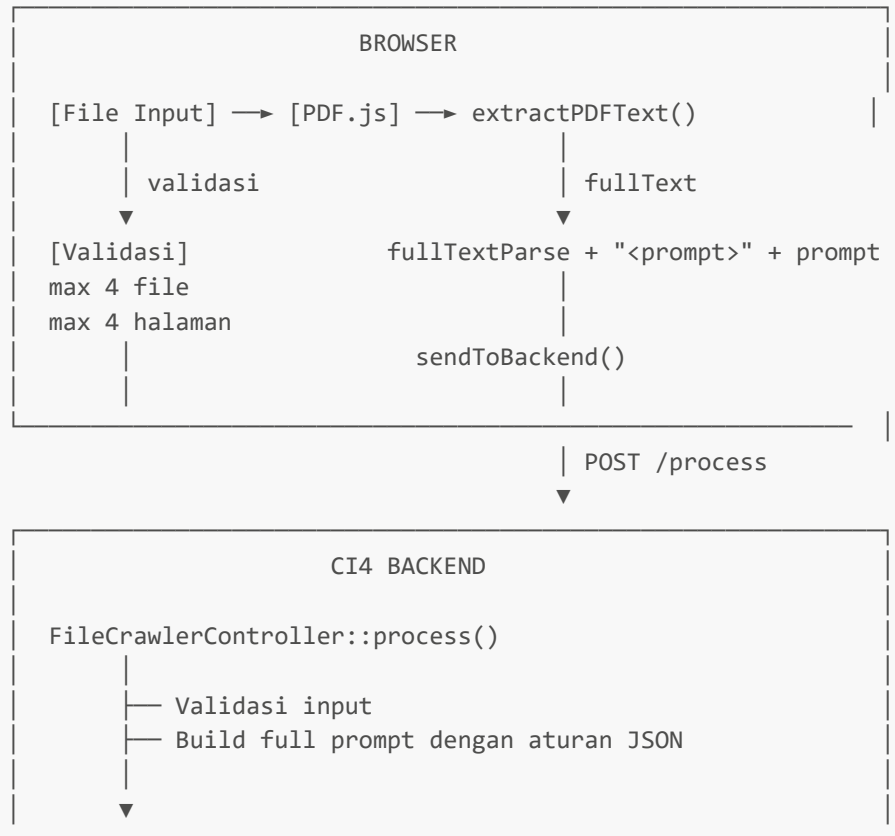
POST `/file-crawler/merge`

```
// Request
{
  "file_ids": ["file-xxxx-xxxx", "file-yyyy-yyyy"],
  "format": "xlsx"
}

// Response sukses
{
  "success": true,
  "merged_id": "merged_67890abcdef",
  "format": "xlsx",
  "file_merge": ["file-xxxx-xxxx", "file-yyyy-yyyy"]
}
```

---

8. Diagram Alur Data



```
│  OpenRouter::complete()                                  │
│       │                                                  │
│       ▼                                                  │
│  [OpenRouter API] ──► [DeepSeek AI]                      │
│       │                                                  │
│       ▼ JSON response                                    │
│  Parse & Validate JSON                                   │
│       │                                                  │
│       ▼                                                  │
│  ScrapeModel::insert() ──► [MySQL: tb_scrape]            │
│       │                                                  │
│       ▼                                                  │
│  Return { success, name, id_file, type }                 │
│                                                          │
└──────────────────────────────────────────────────────────┘
                                  │
                                  ▼ Response
┌──────────────────────────────────────────────────────────┐
│                      BROWSER                             │
│                                                          │
│  allResults.push(result)                                 │
│  updateUI() ──► Render kartu hasil                       │
│                                                          │
│  User actions:                                           │
│    A) Klik download ──► window.location.href = /downloadas/ │
│    B) Centang + Merge ──► POST /merge ──► kartu baru     │
│    C) Flush ──► GET /flush ──► bersihkan UI              │
│                                                          │
└──────────────────────────────────────────────────────────┘
```

---

9. Troubleshooting Umum

AI Response Tidak Valid (JSON Parse Error)

Penyebab: AI mengembalikan teks dengan markdown atau format tidak valid.

Solusi dalam kode:

```php
// Hapus BOM
$response = preg_replace('/^\xEF\xBB\xBF/', '', $response);

// Fix JSON yang terpotong
if (substr($response, -1) === ']') {
    $response .= '}';
}

// Pastikan UTF-8
$response = mb_convert_encoding($response, 'UTF-8', 'UTF-8');
```

Solusi pada prompt: Tambahkan contoh format yang lebih eksplisit di bagian `[ATURAN]`.

Output Buffer Error Saat Download

Penyebab: Ada output (echo, var_dump, dll) sebelum header HTTP dikirim.

Solusi:

```
// Selalu bersihkan buffer sebelum output file
while (ob_get_level()) {
    ob_end_clean();
}
// Baru kirim header dan file
header('Content-Type: ...');
```

File Tidak Muncul Setelah Reload

Penyebab: Cookie `device_fp` belum ada atau berbeda.

Pastikan:

- Cookie `SameSite=Strict` tidak terblokir
- Domain cookie sesuai dengan domain aplikasi
- `getFingerprint()` dipanggil sebelum request ke `get-files`

PhpSpreadsheet / TCPDF Error di CI4

Penyebab: Konflik autoload atau versi PHP.

Solusi:

```
# Update autoload
composer dump-autoload

# Pastikan versi PHP kompatibel
php -v # Minimal PHP 8.0

# Cek instalasi library
composer show | grep -E "tcpdf|phpspreadsheet|phpword"
```

Performance: AI Response Lambat

Penyebab: Model AI besar atau teks input terlalu panjang.

Solusi:

- Batasi jumlah halaman (sudah diimplementasikan: max 4 halaman)
- Gunakan model yang lebih ringan di OpenRouter (misal: `mistral/mistral-7b-instruct:free`)
- Implementasikan timeout yang lebih lama di `OpenRouter.php`:

  ```
  private int $timeout = 120; // Naikkan dari 60 ke 120 detik
  ```

Dokumentasi ini mencakup seluruh komponen FileCrawler.ai dari setup hingga deployment. Setiap bagian dapat dikembangkan secara mandiri sesuai kebutuhan proyek.